**Содержание**

[**Введение** 4](#_Toc106403490)

[**1 Анализ задачи** 6](#_Toc106403491)

[**1.1 Постановка задачи** 6](#_Toc106403492)

[**1.2 Инструменты разработки** 6](#_Toc106403493)

[**1.3 Требования к приложению** 10](#_Toc106403494)

[**2 Проектирование задачи** 12](#_Toc106403495)

[**2.1 Организация данных** 12](#_Toc106403496)

[**2.2 Процессы** 12](#_Toc106403497)

[**2.3 Описание внешнего пользовательского интерфейса** 13](#_Toc106403498)

[**3 Реализация** 14](#_Toc106403499)

[**3.1 Структура программы** 14](#_Toc106403500)

[**3.1.1 Структура и описание процедур и функций пользователя** 14](#_Toc106403501)

[**3.1.2 Описание использованных компонентов** 17](#_Toc106403502)

[**3.2 Спецификация программы** 17](#_Toc106403503)

[**4 Тестирование** 19](#_Toc106403504)

[**5 Руководство пользователя** 21](#_Toc106403505)

[**5.1 Общие сведения о программном продукте** 21](#_Toc106403506)

[**5.2 Инсталляция** 22](#_Toc106403507)

[**5.3 Выполнение программы** 22](#_Toc106403508)

[**5.3.1 Запуск программы** 22](#_Toc106403509)

[**5.3.2 Инструкции по работе с программой** 22](#_Toc106403510)

[**5.3.3 Завершение работы с программой** 27](#_Toc106403511)

[**Заключение** 28](#_Toc106403512)

[**Приложение А** 29](#_Toc106403513)

[**Листинг программы** 29](#_Toc106403514)

# **Введение**

Курсовой проект является завершающим этапом изучения дисциплины «Основы алгоритмизации и программирования». Цель курсового проекта научить учащихся эффективно и рационально использовать разнообразные методы построения программ, применять мощные средства библиотек при решении разнообразных задач.

Целью данного курсового проекта является создание развлекательного игрового приложения «Автогонки». Создаваемое игровое приложение является исключительно развлекательной игрой, предназначенная для пользователей старше 16 лет.

Приведем краткое описание разделов пояснительной записки.

Первый раздел «Анализ задачи» посвящается изучению проблемы в целом, рассмотрению более подробно всех деталей, которые будут использоваться при решении задачи. В этом разделе описывается исследование предметной области, разработки и требования к программе. Также будет рассмотрен и оценён выбор среды разработки программы и будут установлены минимальные требования к аппаратным характеристикам, которые обеспечат правильное выполнение поставленных задач.

Во второй части «Проектирование задачи» описывается логическая и физическая структура данных, внешний пользовательский интерфейс, справочная информация.

«Реализация» - это третья часть пояснительной записки, в которой описываются все элементы и объекты, которые использовались для реализации этого приложения. Здесь предусматривается пошаговое создание форм, а так же спецификация программы, в которой будет представлена полная аннотация файлов используемых в данном проекте.

Раздел четвёртый посвящен «Тестированию». В нём протестирован каждый пункт меню, каждая операция которая выполняется приложением, а также будет предоставлен вывод о результате тестирования.

В разделе пятом «Руководство пользователя» описано назначение приложения, область применения, возможности программы и среда функционирования.

В «Заключении» предоставлена краткая формулировка задачи, проанализированы результаты проделанной работы, описаны использованные методы и средства, степень соответствия проектных решений заданию, возможность модификации.

В «Списке использованных источников» приводится список используемых при разработке источников.

В приложении к пояснительной записке приведен листинг программы с необходимыми комментариями.

Схема работы системы представлена в графической части.

# **1 Анализ задачи**

# **1.1 Постановка задачи**

Темой данного проекта является «Разработка игрового приложения «Автогонки».

При реализации проекта разработчик должен стараться сделать его простым и максимально доступным для пользователей.

Данный проект должен стать общедоступным для большинства пользователей. В поставленной задаче необходимо реализовать максимально простой и пользовательский интерфейс, позволяющий использовать проект пользователю, не обладающему дополнительными знаниями ЭВМ.

Разрабатываемый программный продукт должен позволять выполнять следующие действия:

* просмотр справки;
* тестирование игрового процесса;

В разрабатываемой программе будут использоваться два вида данных: К выходной – Результат игрового процесса. Постоянной информацией в проекте будут являться картинки и gif изображение.

Программный продукт предоставляет функционал только для одного вида пользователя: Игрок.

# **1.2 Инструменты разработки**

Для разработки данного проекта будет выбрана среда **Delphi** так как это одна из мощнейших сред для разработки приложений такого уровня. **Delphi**— это среда объектно-ориентированного программирования, относящаяся к классу RAD – (Rapid Application Development – «Средство быстрой разработки приложений»), основанная на языке Object Pascal. Используется для разработки и поддержки программного обеспечения. В отличии от своего прародителя, языка Pascal, многофункционален и удобен тем, что имеет визуальный редактор приложения, который позволяет создавать внешний облик будущей программы. Приложения Windows для создания которых требовалось большое количество человеческих усилий теперь могут быть написаны одним человеком, использующим Delphi.

Delphi позволяет создать различные виды программ: консольные приложения, оконные приложения, приложения для работы с Интернетом и базами данных. То есть, Delphi является не только средствами для работы с языком программирования Паскаль, но дополнительные инструменты, призванные для максимального упрощения и ускорения создание приложений. К дополнительным инструментам можно отнести визуальный редактор форм, благодаря которому можно с легкостью создать полноценную программу, и другие визуальные составляющие разработки программного обеспечения. С Delphi вам не нужно вручную просчитывать расположение каждого элемента интерфейса пользователя, поэтому при разработке программы значительно экономится время.

Визуальное программирование как бы добавляет новое измерение при создании приложений, давая возможность изображать эти объекты на экране монитора для выполнения самой программы. Без визуального программирования процесс отображения требует написания фрагмента кода, создающего и настраивающего объект «по месту». Увидеть закодированные объекты было возможно только в ходе исполнения программы. При таком подходе достижение того, чтобы объекты выглядели и вели себя заданным образом, становится утомительным процессом, который требует неоднократных исправлений программного кода с последующей прогонкой программы и наблюдения за тем, что в итоге получилось.

Три основные части разработки интерфейса следующие: проектирование панели, проектирование диалога и представление окон. Для общего пользовательского доступа также должны учитываться условия применения архитектуры прикладных систем.

Сегодня появилась реальная возможность с помощью моделирования на современных многофункциональных средствах обработки и отображения информации таких как Delphi конкретизировать тип и характеристики используемых информационных моделей, выявить основные особенности будущей деятельности операторов, сформулировать требования к параметрам аппаратно-программных средств интерфейса взаимодействия и т.д.

 Ни одно серьезное программное обеспечение не обходится без модуля справочной информации и руководства пользователя. Это придает программе законченный вид и показывает заботу о пользователе

**Help & Manual** – легкий в использовании и функциональный инструмент, упрощающий создание справочных файлов Windows, печать справочных руководств и документации в целом. Программа имеет интуитивно понятный интерфейс. Все созданные проекты можно сохранить в различных форматах: HTML Help, Winhelp и MS Help 2.0 / Visual Studio Help, Browser-based Help, PDF и Word RTF, а также печатной документации при использовании одного и того же проекта. В основном окне программы содержатся оглавление (в виде древовидного списка) и текстовый редактор. Это дает возможность легко ориентироваться в оглавлении, редактировать или перемещать разделы справки без каких-либо проблем. Также утилита позволяет конвертировать help-файлы из одного формата в другой. Кроме приложений для работы с текстом в данном продукте содержатся утилиты для создания скриншотов и редактирования графических файлов.

Для создания инсталлятора будет использоваться мощное и удобное средство - **Smart Install Maker**.

Программа обладает удобным и интуитивно понятным интерфейсом, а также полным набором необходимых функций для создания профессиональных инсталляторов с минимальным размером, высокой степенью сжатия файлов и приятным интерфейсом.

Помимо стандартного минимума, Smart Install Maker позволяет редактировать системный реестр и INI-файлы, создавать программные ярлыки, запускать ассоциируемые и исполняемые файлы, регистрировать новые шрифты и ActiveX компоненты, отображать тексты информации и лицензионного соглашения. Также, с помощью этой утилиты, можно создать мультиязыковые инсталляторы с поддержкой более 20-ти популярных языков мира.

Инсталляторы, созданные в среде Smart Install Maker имеют небольшой размер и поддерживают превосходный формат сжатия Cab. Все это позволяет создавать действительно компактные дистрибутивы, которые не создадут проблем при их распространении по интернету.

Особенности Smart Install Maker:

* Современный стиль оформления;
* полностью настраиваемый интерфейс инсталлятора;
* небольшой размер установочного модуля;
* поддержка алгоритма сжатия Cab;
* поддержка мультиязычных инсталляторов;
* интуитивно понятный интерфейс пользователя;
* проверка версии установленного .NET Framework;
* возможность использования переменных директорий;
* возможность разбивки инсталлятора на установочные диски.

**Adobe Photoshop 2021** – редактор для редактирования изображений. Необходим для оформления некоторых элементов интерфейса.

**Microsoft Word 2019** – редактор текста для написания документации.

**Blender** – программа для создания 3d моделей и рендера.

**Delphi 11** – среда программирования.

При создании использовался компьютер со следующими параметрами:

* процессор AMD Ryzen 5 3550H;
* объем ОЗУ 16 Gb;
* объем места на SSD 1 Tb;
* видеокарта Nvidia GeForce RTX 2060;
* ОС – Windows 10.

Как видно разрабатываемое приложение не очень требовательно к аппаратным ресурсам, что, является большим плюсом.

# **1.3 Требования к приложению**

На этапе исследования предметной области был установлен целый ряд требований, которые предъявляются к разрабатываемой программе.

**Требования к интерфейсу:** в связи с частым использованием программы она должна быть с приятной цветовой гаммой и понятной для пользователя.

Следовательно, каждое окно должно иметь ясную визуальную иерархию своих элементов. Фрагменты текста должны располагаться на экране так, чтобы пользователя было просто и понятно принимать информацию.

Пользователь не должен испытывать какого-либо дискомфорта в плане восприятия информация, отображённой на экране. Объекты (рисунки и символы) не должны быть слишком мелкие.

На одной форме нельзя допускать избытка и нагромождения данных.

Формы должны быть эффектно оформлены согласно тематике разрабатываемого проекта.

**Требования к надежности:**

Специальных требований к надежности не предъявляется.

Время восстановления после отказа, вызванного сбоем электропитания технических средств (иными внешними факторами), не фатальным сбоем (не крахом) операционной системы, не должно превышать 30-ти минут при условии соблюдения условий эксплуатации технических и программных средств.

Время восстановления после отказа, вызванного неисправностью технических средств, фатальным сбоем (крахом) операционной системы, не должно превышать времени, требуемого на устранение неисправностей технических средств и переустановки программных средств.

Отказы программы вследствие некорректных действий пользователя при взаимодействии с программой через графический интерфейс не должны влиять на конечный результат.

ПП должен иметь средства фиксации всех действий в процессе изготовления программного продукта. Это необходимо для восстановления состояния процесса разработки на любом его этапе (при комплексной отладке).

**Системные требования:**

- процессор не ниже Pentium 500 Mhz;

- объем ОЗУ не меньше 128 Mb;

- объем места на HDD не менее 200 Mb;

- видеоподсистема не менее 1024х768 точек с глубиной

цвета 16 Bit;

- ОС – Windows 7,8,10.

# **2 Проектирование задачи**

# **2.1 Организация данных**

Проектирование задачи – это очень важный и ответственный этап в разработке любого приложения. Важным является он вследствие того, что методы, по средствам которых пользователь управляет формами, построены на высокой степени специализации каждого из компонентов. Необходимым условием при разработке данного приложения является описание организации данных, т.е. логическая и физическая структура данных в контексте среды разработки. В разрабатываемой программе будут использоваться три вида данных. Первым видом являются данные, которые будут введены разработчиком на этапе реализации задачи. Сюда можно отнести изображения (иконки), описание, исходные коды сортировок. Вторым видом данных, используемых в программе, является вводимая пользователем информация. Тип данных: string. Третьим видом данных является результат программы. Его также относят к отдельному виду, так как ни пользователь, ни разработчик его не вводят, а программа сама получает его в результате выполнения определенных действий. Таким образом, организация данных является важной задачей при разработке данной и любой программы.

# **2.2 Процессы**

Согласно всем перечисленным требованиям и указаниям, которые были рассмотрены в разделе «Анализ задачи», было определено, чем конкретно должна заниматься разрабатываемая курсовая программа. Главной ее задачей будет являться: получение удовольствия от процесса игры, к примеру, во время досуга или же в целях скоротать время. Весь данный процесс будет заключаться в следующем. Пользователь управляя красной машинкой обгоняет другие машины, способом перемещения нажимая ЛКМ (движение влево) и ПКМ (движение вправо), при этом набирая за это очки. С каждой новой игрой пользователь будет стремиться набрать как можно больше очков.

# **2.3 Описание внешнего пользовательского интерфейса**

Важным при выполнении курсового проекта является организация диалога между, пользователем и самой программой. Во многом это зависит от того, как программист разработает данную программу, какие компоненты будут использованы и какие методы будут автоматизированы. Во-первых, особое внимание следует уделить интерфейсу. Разработчик должен так организовать внешний вид своей программы, чтобы пользователь понял, что от него требуется. Для организации эффективной работы пользователя нужно создать целостное приложение данной предметной области, в которой все компоненты приложения будут сгруппированы по функциональному назначению. При этом необходимо обеспечить удобный графический интерфейс пользователя. Приложение должно позволить пользователю решать задачи, затрачивая значительно меньше усилий, чем при работе с разрозненными объектами.

Ниже представлена система меню и организация навигации между окнами программы (Рисунок 1):

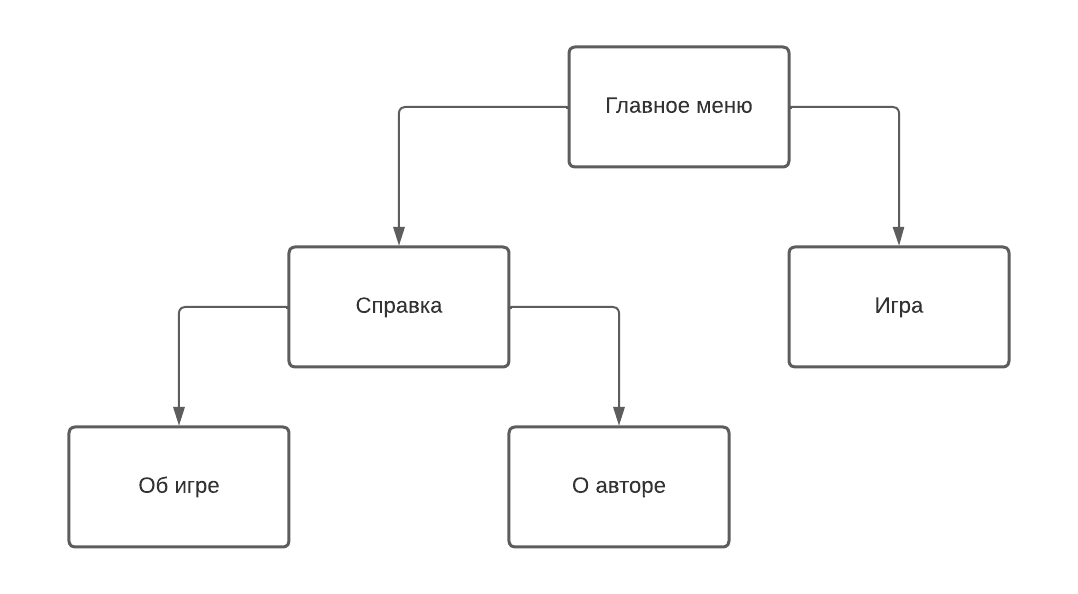


Рисунок 1 – Навигация между окнами программы.

# **3 Реализация**

# **3.1 Структура программы**

Данный курсовой проект содержит 4 модулей. Далее рассмотрим назначение каждого модуля:

1. Модуль Unit1 - отвечает за навигацию по программе (Главное окно). На ней же можно просмотреть данные об игре;
2. Модуль Unit3 – открывается при нажатии на кнопку «Запустить игру» и содержит сам игровой процесс;
3. Модуль Unit4 – представляет собой окно с информацией о авторе, запускается в главном окне игрового приложения;
4. Модуль Unit5 – представляет собой заставку игрового приложения.

# **3.1.1 Структура и описание процедур и функций пользователя**

Описание разработанных процедур находятся в таблице 1.

Таблица 1 – Процедуры и функции

|  |  |  |  |
| --- | --- | --- | --- |
| Имя процедуры (функции) | В каком  модуле  находится | За каким  компонентом закреплена | Назначение |
| 1 | 2 | 3 | 4 |
| procedure TForm1.Button1Click(Sender: TObject); | Unit1 | Button1 | Переход на форму с игрой |

Продолжение таблицы 1 – Процедуры и функции

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| procedure TForm1.Button2Click(Sender: TObject); | Unit1 | Button2 | Закрывает игру |
| procedure TForm1.FormCreate(Sender: TObject); | Form1 | Загружает gif файл |
| procedure TForm1.N6Click(Sender: TObject); | MainMenu1 | Переход на форму “об авторе” |
| procedure TForm1.N5Click(Sender: TObject); | MainMenu1 | Открывает справку |
| procedure TForm1.N4Click(Sender: TObject); | MainMenu1 | Закрывает игру |
| procedure TForm1.N2Click(Sender: TObject); | MainMenu1 | Переход на форму с игрой |
| procedure TForm4.Button1Click(Sender: TObject); | Unit4 | Button1 | Переход в главное меню |
| procedure TForm5.Timer1Timer(Sender: TObject); | Unit5 | Timer1 | Переход в главное меню спустя определённое время |
| procedure FormMouseDown(Sender: TObject; Button: TMouseButton;  Shift: TShiftState; X, Y: Integer); | Unit3 | MouseButton | Передвижение игрока кнопками мыши |
| procedure Timer1Timer(Sender: TObject); | Timer1 | Появление машины 1 спустя определенное время |
| procedure FormCreate(Sender: TObject); | Form3 | Выполняет случайную позицию появление машин |
| procedure Button1Click(Sender: TObject); | Button1 | Запуск игрового процесса |
| procedure Timer2Timer(Sender: TObject); | Timer2 | Появление машины 2 спустя определенное время |
| procedure Timer3Timer(Sender: TObject); | Timer3 | Появление машины 3 спустя определенное время |
| procedure Timer4Timer(Sender: TObject); | Timer4 | Появление машины 4 спустя определенное время |

Продолжение таблицы 1 – Процедуры и функции

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| procedure Timer5Timer(Sender: TObject); | Unit3 | Timer5 | Появление машины 5 спустя определенное время |
| procedure Timer6Timer(Sender: TObject); | Timer6 | Появление машины 6 спустя определенное время |
| procedure Timer7Timer(Sender: TObject); | Timer7 | Появление машины 7 спустя определенное время |
| procedure Timer8Timer(Sender: TObject); | Timer8 | Появление машины 8 спустя определенное время |
| procedure Timer9Timer(Sender: TObject); | Timer9 | Появление машины 9 спустя определенное время |
| procedure Timer10Timer(Sender: TObject); | Timer10 | Появление машины 10 спустя определенное время |
| procedure Button3Click(Sender: TObject); | Button3 | Приостановка игрового процесса |
| procedure Button2Click(Sender: TObject); | Button2 | Закрытие игры |
| procedure Button4Click(Sender: TObject); | Button4 | Переход в главное меню |

# 

# **3.1.2 Описание использованных компонентов**

Описание использованных для разработки приложения компонентов приводится в таблице 2.

Таблица 2 – Использованные компоненты

|  |  |  |
| --- | --- | --- |
| Компонент | На какой форме  расположен | Назначение |
| TImage | Form 3- 5 | Используется как фон |
| TMainMenu | Form 1 | Используется для добавления новых вкладок в окне |
| TLabel | Form 4 | Отображение надписей на форме |

Продолжение таблицы – Использованные компоненты

|  |  |  |
| --- | --- | --- |
| Компонент | На какой форме расположен | Назначение |
| TButton | Form 1,3-4 | Используется для взаимодействия пользователя с программой: перейти на форму, узнать результат теста |
| TTimer | Form 3,5 | Используется для отсчета интервалов реального времени |

# **3.2 Спецификация программы**

Точное название проекта и его состав приводится в Таблице 3.

Таблица 3 – Спецификация программы

|  |  |
| --- | --- |
| Имя файла | Назначение |
| 1 | 2 |
| 1. Project1 | Исполняемый файл проекта |
| 2. Project1.dpr | Файл проекта, связывает все файлы, из которых состоит приложение |
| 3. Unit 1,3-5 | Файлы программного модуля для форм |
| 4. Setup.exe | Установочный файл |
| 5. Справка.chm | Файл справки, содержит помощь по работе с программой |
| 6. Unit1.dfm | Форма главного меню |
| 7. Unit3.dfm | Форма с основной игрой |
| 8. Unit4.dfm | Формы с информацией о авторе |
| 9. Unit5.dfm | Форма с заставкой |

# **4 Тестирование**

При разработке данной программы многие возникающие ошибки и недоработки были исправлены на этапе реализации проекта. После завершения испытания реализации программы было проведено тщательное функциональное тестирование. Функциональное тестирование должно гарантировать работу всех элементов программы в автономном режиме.

Отчёт о результатах тестирования предоставлен в Таблице 4.

Таблица 4 – Отчёт о результатах тестирования

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Тест | | Ожидаемый  результат | | | Физический  результат | Результат  тестирования | |
| 1. Проверка кнопки «Выход» | | Закрытие игрового приложения | | | | Выполнено | |
| 2. Проверка кнопки «Запустить игру» | | Переход на форму с основной игрой | Осуществляется переход на форму с основной игрой | | | Выполнено | |
| 3. Проверка кнопки «Начать игру» | | Запуск игрового процесса | | | | Выполнено | |
| 4. Проверка кнопки «Пауза» | | Приостанавливает игровой процесс | | | | Выполнено | |
| 5. Проверка пункта меню «Завершение игры» | | Закрытие игрового приложения | | | | Выполнено | |
| 6. Проверка пункта Справка «О авторе» | Переход к форме с информации о авторе | | | Осуществляется переход к форме с информации о авторе | | Выполнено | |
| 7. Проверка пункта Справка «Об игре» | Открытие справки об игре | | | | | Выполнено | |
| 8 Проверка пункта Меню «Выход» | Закрытие игрового приложения | | | | | Выполнено | |
| 9 Проверка пункта Меню «Запустить игру» | Переход на форму с основной игрой | | Осуществляется переход на форму с основной игрой | | | Выполнено | |
| 10. Проверка кнопки «Главное меню» | Переход на главную форму | | Осуществляется переход на главную форму | | | Выполнено | |

Элементы программы были проверены, и было установлено, что все они работают правильно и выполняют задачи, указанные в процедурах.

При использовании данного игрового приложения пользователь не столкнётся с ошибками системы.

# **5 Руководство пользователя**

# **5.1 Общие сведения о программном продукте**

Разработанная программа является игровым приложением, которая представляет собой исключительно развлекательную игру.

Данный программный продукт предназначен для любых видов пользователей достигших 16 лет.

Быстродействие любой программы во многом зависит от характеристик выбранного персонального компьютера: рабочей частоты процессора, объема оперативной памяти и т.д. Несмотря на все реализованные в ней задачи, она легко запускается и функционирует на любых машинах.

Тестирование проводилось на разных классах ЭВМ, и работать с данной программой было комфортно. Программа разработана на ПК со следующими характеристиками:

- процессор AMD Ryzen 5 3550H;

- объем ОЗУ 16 Gb;

- объем места на SSD 1 Tb;

- видеокарта Nvidia GeForce RTX 2060;

- ОС – Windows 10.

# **5.2 Инсталляция**

Для того чтобы установить программу необходимо запустить файл Setup.exe. Появится окно установки игрового приложения “Автогонки”. Следуем инструкции и устанавливаем приложение.

# **5.3 Выполнение программы**

# **5.3.1 Запуск программы**

Данное игровое приложение можно запустить следующим способом.

Способом является запуск из каталога, в который устанавливалось приложение (по умолчанию C:\Program Files\Автогонки).

По подготовленным тестам будет осуществляться функциональное и полное тестирование программного продукта. Отчет о результатах тестирования будет представлен в 4 разделе пояснительной записки**.**

# **5.3.2 Инструкции по работе с программой**

При запуске игрового приложения Пользователя встречает заставка игры

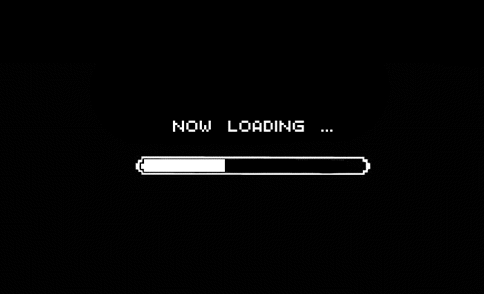


Рисунок 2 – Загрузочный экран

После заставки появляется главное меню

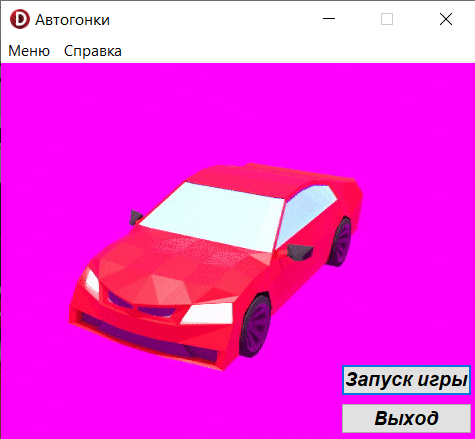


Рисунок 3 – Главное меню

В главном меню находятся 2 основные кнопки: "Запуск игры" и "Выход". Помимо этих кнопок есть вкладки: "Меню" и "Справка". В пункте "Меню" находятся кнопки "Запустить игру" и "Выход". А в пункте "Справка" находятся две следующие кнопки: "Об игре" и "О авторе". Нажав кнопку "Об игре" пользователю предоставится справка об игре, которую вы сейчас читаете. А нажав кнопку "О авторе" откроется окно в котором вы знакомитесь с разработчиком игры. В этом же окне есть кнопка для возвращения в главное меню - "Главное меню".

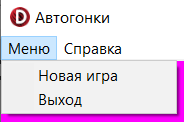


Рисунок 4 - Меню

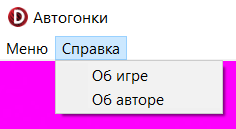


Рисунок 5 - Справка

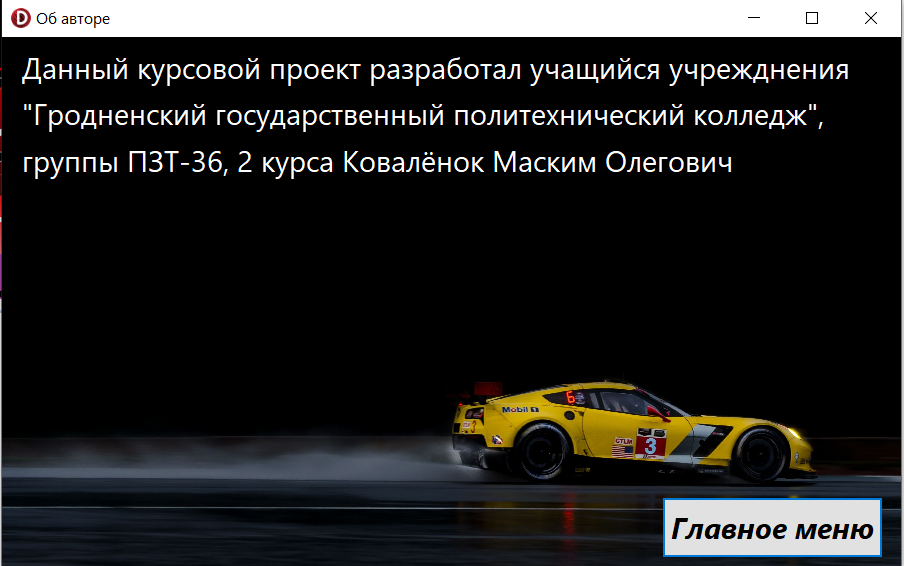


Рисунок 6 – Об авторе

При нажатии Кнопки "Выход" и "Выход" во вкладке "Меню" производится выход из игрового приложения. А при нажатии "Запуск игры" и "Запустить игру" во вкладке "Меню" появляется новое окно с самой игрой.

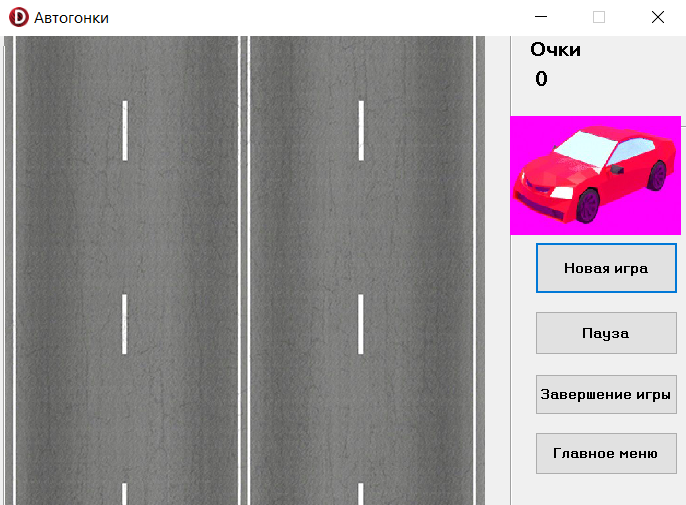


Рисунок 7 – Основное окно игры

В данном окне есть 4 кнопки: "Новая игра", "Пауза", "Завершение игры" и "Главное меню". Нажав на кнопку "Главное меню" пользователя переносит на главное меню. Нажав на кнопку "Завершение игры" производиться закрытие игры. нажав на кнопку "Новая игра" запускает игровой процесс в котором пользователь будет управлять красной машинкой. Задача пользователя объезжать другие машины используя кнопки передвижения - ЛКМ и ПКМ. ЛКМ - движение влево. ПКМ - движение вправо. За каждую машину которую пользователь обогнал по горизонтальной оси зачисляется +10 очков. Нажав на кнопку "Пауза" игра приостанавливается.

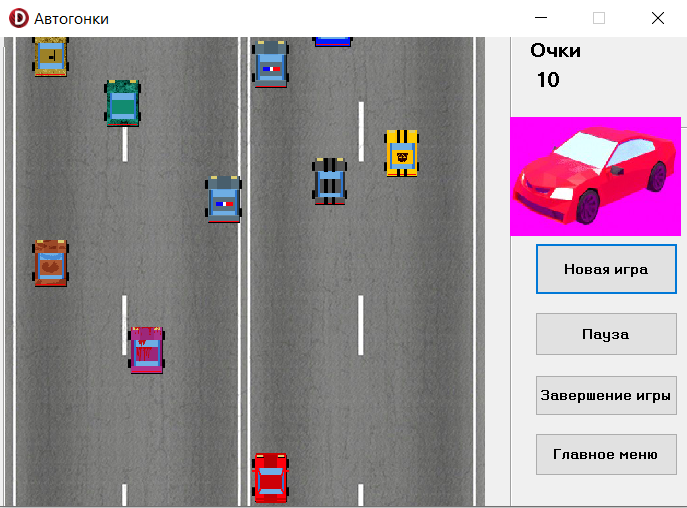


Рисунок 8 – Процесс игры

При столкновении игрока с другой машиной игра завершается (т. е. пользователь потерпел неудачу), выводя аналогичное сообщение "Вы врезались".

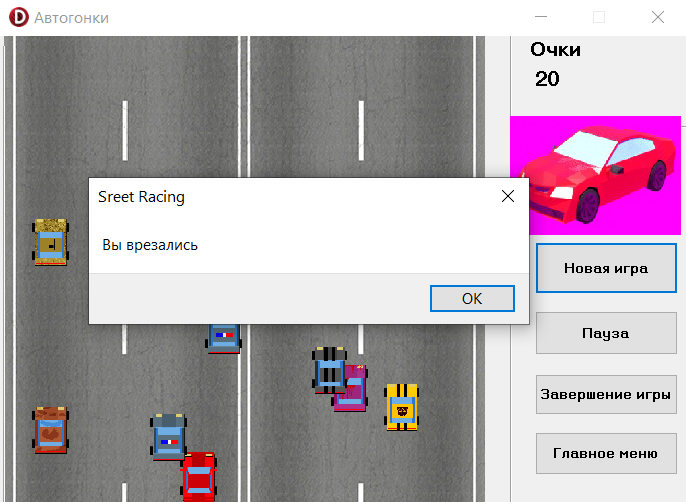


Рисунок 9 – Конец игры

# **5.3.3 Завершение работы с программой**

Завершить работу с программой можно несколькими способами.

Одним из них является закрытие через меню. Нужно нажать на крестик в углу формы.

Также можно закрыть приложение, нажав пункт меню "Выход".

Справочную систему можно запустить с помощью пункта справка "Об игре".

# **Заключение**

Целью данного проекта было разработать игровое приложение “Автогонки”.

В поставленной задаче был реализован простой и понятный пользовательский интерфейс.

В ходе тестирования все исключительные ситуации были обработаны. Проект работает без сбоев и ошибок.

В процессе разработки программного продукта я научился создавать динамические компоненты на форме, проработал их взаимодействие, закрепил умение создания собственных процедур, научился обрабатывать все исключительные ситуации.

Продукт имеет дальнейшее развитие, так как на данный момент разнообразие и возможности весьма малы. Можно добавить в сам игровой процесс уровни и сложность. Это могло бы сделать программный продукт более интересным и масштабным. Несмотря на все это, данный программный продукт имеет законченный вид.

Исходя из этого, можно сделать вывод, что программа реализована успешно.

# **Приложение А**

# **Листинг программы**

unit Unit1;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, ExtCtrls, StdCtrls, jpeg, Vcl.Imaging.pngimage, ShellApi, Gifimg, Vcl.Menus;

type

TForm1 = class(TForm)

Image1: TImage;

MainMenu1: TMainMenu;

N1: TMenuItem;

N2: TMenuItem;

N4: TMenuItem;

Button1: TButton;

Button2: TButton;

N3: TMenuItem;

N5: TMenuItem;

N6: TMenuItem;

procedure Button1Click(Sender: TObject);

procedure Button2Click(Sender: TObject);

procedure N2Click(Sender: TObject);

procedure N4Click(Sender: TObject);

procedure N6Click(Sender: TObject);

procedure N3Click(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure N5Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form1: TForm1;

GIF : TGIFImage;

implementation

{$R \*.dfm}

uses Unit3, Unit2, Unit5, Unit4;

procedure TForm1.Button1Click(Sender: TObject);

begin

form3.visible:=True;

form1.visible:=False;

end;

procedure TForm1.Button2Click(Sender: TObject);

begin

form5.close;

close;

end;

procedure TForm1.FormCreate(Sender: TObject);

begin

GIF := TGIFImage.Create;

GIF.LoadFromFile('ezgif.com-gif-maker.gif');

GIF.Animate :=True;

Image1.Picture.Assign(GIF);

end;

procedure TForm1.N2Click(Sender: TObject);

begin

form3.visible:=True;

form1.visible:=False;

end;

procedure TForm1.N3Click(Sender: TObject);

begin

shellexecute(handle,nil,'Справка.chm',nil,nil,sw\_restore);

end;

procedure TForm1.N4Click(Sender: TObject);

begin

form5.close;

Close;

end;

procedure TForm1.N5Click(Sender: TObject);

begin

shellexecute(handle,nil,'Справка.chm',nil,nil,sw\_restore);

end;

procedure TForm1.N6Click(Sender: TObject);

begin

Form4.Visible:=True;

Form1.Visible:=False;

end;

end.

unit Unit3;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, ExtCtrls, StdCtrls, jpeg, Vcl.Imaging.pngimage;

type

TForm3 = class(TForm)

Block1: TImage;

Car: TImage;

Block2: TImage;

Block3: TImage;

Block9: TImage;

Block10: TImage;

Block7: TImage;

Block8: TImage;

Block6: TImage;

Block4: TImage;

Block5: TImage;

Timer1: TTimer;

Label1: TLabel;

Button1: TButton;

Label2: TLabel;

Timer2: TTimer;

Timer3: TTimer;

Bevel1: TBevel;

Bevel2: TBevel;

Bevel3: TBevel;

Image1: TImage;

Timer4: TTimer;

Timer5: TTimer;

Timer6: TTimer;

Timer7: TTimer;

Timer8: TTimer;

Timer9: TTimer;

Timer10: TTimer;

Button3: TButton;

Image2: TImage;

Button2: TButton;

Button4: TButton;

procedure FormMouseDown(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

procedure Timer1Timer(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure Button1Click(Sender: TObject);

procedure Timer2Timer(Sender: TObject);

procedure Timer3Timer(Sender: TObject);

procedure Timer4Timer(Sender: TObject);

procedure Timer5Timer(Sender: TObject);

procedure Timer6Timer(Sender: TObject);

procedure Timer7Timer(Sender: TObject);

procedure Timer8Timer(Sender: TObject);

procedure Timer9Timer(Sender: TObject);

procedure Timer10Timer(Sender: TObject);

procedure Button3Click(Sender: TObject);

procedure Button2Click(Sender: TObject);

procedure Button4Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form3: TForm3;

Bonus:integer;

implementation

{$R \*.dfm}

uses Unit1, Unit5;

procedure GameOver;

begin

form3.Timer1.Enabled:=false;

form3.Timer2.Enabled:=false;

form3.Timer3.Enabled:=false;

form3.Timer4.Enabled:=false;

form3.Timer5.Enabled:=false;

form3.Timer6.Enabled:=false;

form3.Timer7.Enabled:=false;

form3.Timer8.Enabled:=false;

form3.Timer9.Enabled:=false;

form3.Timer10.Enabled:=false;

ShowMessage('Вы врезались');

end;

procedure TForm3.FormMouseDown(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

begin

If (Button=mbLeft) and (Car.Left>10) then

Car.Left:=Car.Left-(Car.Width div 2);

If (Button=mbRight) and (Car.Left<(465-Car.Width)) then

Car.Left:=Car.Left+(Car.Width div 2);

end;

procedure TForm3.Timer1Timer(Sender: TObject);

begin

Label1.Caption:=IntToStr(Bonus);

Block1.Top:=Block1.Top+(Block1.Height div 2);

if (Block1.Top+Block1.Height)>(Car.Top) then

if (Block1.Left+1<=Car.Left) and (Car.Left<=(Block1.Left+Block1.Width-1))

then GameOver;

if (Block1.Top+Block1.Height)>(Car.Top) then

if (Block1.Left+1<=(Car.Left+Car.Width)) and ((Car.Left+Car.Width)<=(Block1.Left+Block1.Width-1))

then GameOver;

If Block1.Top+Block1.Height>=(Form3.ClientHeight) then

begin

Bonus:=Bonus+10;

Block1.Hide;

Block1.Top:=1;

repeat

Block1.Left:=Random(465-Car.Width);

until (Block1.Left>10);

Block1.Show;

end;

end;

procedure TForm3.FormCreate(Sender: TObject);

begin

Randomize;

end;

procedure TForm3.Button1Click(Sender: TObject);

begin

Block1.Left:=128;

Block1.Top:=-288;

Block2.Left:=384;

Block2.Top:=-216;

Block3.Left:=32;

Block3.Top:=-8;

Block4.Left:=205;

Block4.Top:=-32;

Block5.Left:=400;

Block5.Top:=-8;

Block6.Left:=312;

Block6.Top:=-112;

Block7.Left:=104;

Block7.Top:=-120;

Block8.Left:=208;

Block8.Top:=-176;

Block9.Left:=312;

Block9.Top:=-280;

Block10.Left:=32;

Block10.Top:=-240;

Car.Left:=216;

Car.Top:=415;

Bonus:=0;

Label1.Caption:='0';

Block1.Show;

Block2.Show;

Block3.Show;

Block4.Show;

Block5.Show;

Block6.Show;

Block7.Show;

Block8.Show;

Block9.Show;

Block10.Show;

Car.Show;

Timer1.Enabled:=true;

Timer2.Enabled:=true;

Timer3.Enabled:=true;

Timer4.Enabled:=true;

Timer5.Enabled:=true;

Timer6.Enabled:=true;

Timer7.Enabled:=true;

Timer8.Enabled:=true;

Timer9.Enabled:=true;

Timer10.Enabled:=true;

end;

procedure TForm3.Timer2Timer(Sender: TObject);

begin

Label1.Caption:=IntToStr(Bonus);

Block2.Top:=Block2.Top+(Block2.Height div 3);

if (Block2.Top+Block2.Height)>(Car.Top) then

if (Block2.Left+1<=Car.Left) and (Car.Left<=(Block2.Left+Block2.Width-1))

then GameOver;

if (Block2.Top+Block2.Height)>(Car.Top) then

if (Block2.Left+1<=(Car.Left+Car.Width)) and ((Car.Left+Car.Width)<=(Block2.Left+Block2.Width-1))

then GameOver;

If Block2.Top+Block2.Height>=(Form3.ClientHeight) then

begin

Bonus:=Bonus+10;

Block2.Hide;

Block2.Top:=1;

repeat

Block2.Left:=Random(465-Car.Width);

until (Block2.Left>10);

Block2.Show;

end;

end;

procedure TForm3.Timer3Timer(Sender: TObject);

begin

Label1.Caption:=IntToStr(Bonus);

Block3.Top:=Block3.Top+(Block3.Height div 6);

if (Block3.Top+Block3.Height)>(Car.Top) then

if (Block3.Left+1<=Car.Left) and (Car.Left<=(Block3.Left+Block3.Width-1))

then GameOver;

if (Block3.Top+Block3.Height)>(Car.Top) then

if (Block3.Left+1<=(Car.Left+Car.Width)) and ((Car.Left+Car.Width)<=(Block3.Left+Block3.Width-1))

then GameOver;

If Block3.Top+Block3.Height>=(form3.ClientHeight) then

begin

Bonus:=Bonus+10;

Block3.Hide;

Block3.Top:=1;

repeat

Block3.Left:=Random(465-Car.Width);

until (Block3.Left>10);

Block3.Show;

end;

end;

procedure TForm3.Timer4Timer(Sender: TObject);

begin

Label1.Caption:=IntToStr(Bonus);

Block4.Top:=Block4.Top+(Block4.Height div 4);

if (Block4.Top+Block4.Height)>(Car.Top) then

if (Block4.Left+1<=Car.Left) and (Car.Left<=(Block4.Left+Block4.Width-1))

then GameOver;

if (Block4.Top+Block4.Height)>(Car.Top) then

if (Block4.Left+1<=(Car.Left+Car.Width)) and ((Car.Left+Car.Width)<=(Block4.Left+Block4.Width-1))

then GameOver;

If Block4.Top+Block4.Height>=(form3.ClientHeight) then

begin

Bonus:=Bonus+10;

Block4.Hide;

Block4.Top:=1;

repeat

Block4.Left:=Random(465-Car.Width);

until (Block4.Left>10);

Block4.Show;

end;

end;

procedure TForm3.Timer5Timer(Sender: TObject);

begin

Label1.Caption:=IntToStr(Bonus);

Block5.Top:=Block5.Top+(Block5.Height div 3);

if (Block5.Top+Block5.Height)>(Car.Top) then

if (Block5.Left+1<=Car.Left) and (Car.Left<=(Block5.Left+Block5.Width-1))

then GameOver;

if (Block5.Top+Block5.Height)>(Car.Top) then

if (Block5.Left+1<=(Car.Left+Car.Width)) and ((Car.Left+Car.Width)<=(Block5.Left+Block5.Width-1))

then GameOver;

If Block5.Top+Block5.Height>=(form3.ClientHeight) then

begin

Bonus:=Bonus+10;

Block5.Hide;

Block5.Top:=1;

repeat

Block5.Left:=Random(465-Car.Width);

until (Block5.Left>10);

Block5.Show;

end;

end;

procedure TForm3.Timer6Timer(Sender: TObject);

begin

Label1.Caption:=IntToStr(Bonus);

Block6.Top:=Block6.Top+(Block6.Height div 5);

if (Block6.Top+Block6.Height)>(Car.Top) then

if (Block6.Left+1<=Car.Left) and (Car.Left<=(Block6.Left+Block6.Width-1))

then GameOver;

if (Block6.Top+Block6.Height)>(Car.Top) then

if (Block6.Left+1<=(Car.Left+Car.Width)) and ((Car.Left+Car.Width)<=(Block6.Left+Block6.Width-1))

then GameOver;

If Block6.Top+Block6.Height>=(form3.ClientHeight) then

begin

Bonus:=Bonus+10;

Block6.Hide;

Block6.Top:=1;

repeat

Block6.Left:=Random(465-Car.Width);

until (Block6.Left>10);

Block6.Show;

end;

end;

procedure TForm3.Timer7Timer(Sender: TObject);

begin

Label1.Caption:=IntToStr(Bonus);

Block7.Top:=Block7.Top+(Block7.Height div 6);

if (Block7.Top+Block7.Height)>(Car.Top) then

if (Block7.Left+1<=Car.Left) and (Car.Left<=(Block7.Left+Block7.Width-1))

then GameOver;

if (Block7.Top+Block7.Height)>(Car.Top) then

if (Block7.Left+1<=(Car.Left+Car.Width)) and ((Car.Left+Car.Width)<=(Block7.Left+Block7.Width-1))

then GameOver;

If Block7.Top+Block7.Height>=(form3.ClientHeight) then

begin

Bonus:=Bonus+10;

Block7.Hide;

Block7.Top:=1;

repeat

Block7.Left:=Random(465-Car.Width);

until (Block7.Left>10);

Block7.Show;

end;

end;

procedure TForm3.Timer8Timer(Sender: TObject);

begin

Label1.Caption:=IntToStr(Bonus);

Block8.Top:=Block8.Top+(Block8.Height div 2);

if (Block8.Top+Block8.Height)>(Car.Top) then

if (Block8.Left+1<=Car.Left) and (Car.Left<=(Block8.Left+Block8.Width-1))

then GameOver;

if (Block8.Top+Block8.Height)>(Car.Top) then

if (Block8.Left+1<=(Car.Left+Car.Width)) and ((Car.Left+Car.Width)<=(Block8.Left+Block8.Width-1))

then GameOver;

If Block8.Top+Block8.Height>=(form3.ClientHeight) then

begin

Bonus:=Bonus+10;

Block8.Hide;

Block8.Top:=1;

repeat

Block8.Left:=Random(465-Car.Width);

until (Block8.Left>10);

Block8.Show;

end;

end;

procedure TForm3.Timer9Timer(Sender: TObject);

begin

Label1.Caption:=IntToStr(Bonus);

Block9.Top:=Block9.Top+(Block9.Height div 4);

if (Block9.Top+Block9.Height)>(Car.Top) then

if (Block9.Left+1<=Car.Left) and (Car.Left<=(Block9.Left+Block9.Width-1))

then GameOver;

if (Block9.Top+Block9.Height)>(Car.Top) then

if (Block9.Left+1<=(Car.Left+Car.Width)) and ((Car.Left+Car.Width)<=(Block9.Left+Block9.Width-1))

then GameOver;

If Block9.Top+Block9.Height>=(form3.ClientHeight) then

begin

Bonus:=Bonus+10;

Block9.Hide;

Block9.Top:=1;

repeat

Block9.Left:=Random(465-Car.Width);

until (Block9.Left>10);

Block9.Show;

end;

end;

procedure TForm3.Timer10Timer(Sender: TObject);

begin

Label1.Caption:=IntToStr(Bonus);

Block10.Top:=Block10.Top+(Block10.Height div 5);

if (Block10.Top+Block10.Height)>(Car.Top) then

if (Block10.Left+1<=Car.Left) and (Car.Left<=(Block10.Left+Block10.Width-1))

then GameOver;

if (Block10.Top+Block10.Height)>(Car.Top) then

if (Block10.Left+1<=(Car.Left+Car.Width)) and ((Car.Left+Car.Width)<=(Block10.Left+Block10.Width-1))

then GameOver;

If Block10.Top+Block10.Height>=(form3.ClientHeight) then

begin

Bonus:=Bonus+10;

Block10.Hide;

Block10.Top:=1;

repeat

Block10.Left:=Random(465-Car.Width);

until (Block10.Left>10);

Block10.Show;

end;

end;

procedure TForm3.Button2Click(Sender: TObject);

begin

if Button3.Caption='Пауза' then

begin

Timer1.Enabled:=false;

Timer2.Enabled:=false;

Timer3.Enabled:=false;

Timer4.Enabled:=false;

Timer5.Enabled:=false;

Timer6.Enabled:=false;

Timer7.Enabled:=false;

Timer8.Enabled:=false;

Timer9.Enabled:=false;

Timer10.Enabled:=false;

end;

form5.close;

form3.Close;

end;

procedure TForm3.Button3Click(Sender: TObject);

begin

if Button3.Caption='Пауза' then

begin

Timer1.Enabled:=false;

Timer2.Enabled:=false;

Timer3.Enabled:=false;

Timer4.Enabled:=false;

Timer5.Enabled:=false;

Timer6.Enabled:=false;

Timer7.Enabled:=false;

Timer8.Enabled:=false;

Timer9.Enabled:=false;

Timer10.Enabled:=false;

Button3.Caption:='Вперед';

exit

end;

if Button3.Caption='Вперед' then

begin

Timer1.Enabled:=true;

Timer2.Enabled:=true;

Timer3.Enabled:=true;

Timer4.Enabled:=true;

Timer5.Enabled:=true;

Timer6.Enabled:=true;

Timer7.Enabled:=true;

Timer8.Enabled:=true;

Timer9.Enabled:=true;

Timer10.Enabled:=true;

Button3.Caption:='Пауза';

end;

end;

procedure TForm3.Button4Click(Sender: TObject);

begin

form1.visible:=True;

form3.visible:=False;

end;

end.

unit Unit4;

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.StdCtrls, Vcl.Imaging.jpeg,

Vcl.ExtCtrls;

type

TForm4 = class(TForm)

Image1: TImage;

Label1: TLabel;

Label2: TLabel;

Button1: TButton;

Label3: TLabel;

procedure Button1Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form4: TForm4;

implementation

{$R \*.dfm}

uses Unit1, Unit3, Unit5;

procedure TForm4.Button1Click(Sender: TObject);

begin

Form1.Visible:=True;

Form4.Visible:=False;

end;

end.

unit Unit5;

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.ExtCtrls, Vcl.Imaging.pngimage, Gifimg;

type

TForm5 = class(TForm)

Timer1: TTimer;

Image1: TImage;

procedure Timer1Timer(Sender: TObject);

procedure FormCreate(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form5: TForm5;

GIF : TGIFImage;

implementation

{$R \*.dfm}

uses Unit1;

procedure TForm5.FormCreate(Sender: TObject);

begin

GIF := TGIFImage.Create;

GIF.LoadFromFile('6tIB.gif');

GIF.Animate :=True;

Image1.Picture.Assign(GIF);

end;

procedure TForm5.Timer1Timer(Sender: TObject);

begin

timer1.Enabled:=false;

form5.visible:=false;

form1.visible:=true;

end;

end.