Учреждение образования

«Гродненский государственный политехнический колледж»

Специальность: 2 – 40 01 01 «Программное обеспечение информационных

технологий»

Специализация: 2 – 40 01 01 33 «Компьютерная графика»

Предмет: «Основы алгоритмизации и программирования»

Группа: ПЗТ – 41

**КУРСОВОЙ ПРОЕКТ**

Тема: «Разработка игрового приложения «Пиксель»

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

Разработала А. А. Кузьмицкая

Руководитель проекта Т. И. Шатова

2024

Учреждение образования «Гродненский государственный политехнический колледж»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| УТВЕРЖДАЮ | | | | | |
| Председатель цикловой комиссии | | | | | |
|  |  | В.Д. Орехво | | | |
| (подпись) |  | (инициалы, фамилия) | | | |
|  | | | 20 |  | г. |

**ЗАДАНИЕ**

**по курсовому проекту (курсовой работе)**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Учащемуся(имся) | | | | Кузьмицкой Анжелике Александровне | | | | | | | | | | | | | | |
| .(фамилия, собственное имя, отчество (если таковое имеется) | | | | | | | | | | | | | | | | | | |
| курса | 2 | | | | | учебной группы | | | ПЗТ-41 | | | | | | | | | |
| по учебному предмету, модулю | | | | | | Основы алгоритмизации и программирования | | | | | | | | | | | | |
| Тема курсового проекта (курсовой работы) | | | | | | «Разработка игрового приложения «Пиксель» | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | |
| Исходные данные | | | | | |  | | | | | | | | | | | | |
| правила и стратегии игр, аудиофайлы | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | |
| **Содержание курсового проекта (курсовой работы)** | | | | | | | | | | | | | | | | | | |
| **Пояснительная записка** | | | | | | | | | | | | | | | | | | |
| Содержание разделов | | | | | | | | | | | | | | Срок выполнения | | | | |
| Введение. 1 Анализ задачи | | | | | | | | | | | | | | 29.05.2024 | | | | |
| 2 Проектирование задачи | | | | | | | | | | | | | | 31.05.2024 | | | | |
| 3 Реализация | | | | | | | | | | | | | | 10.06.2024 | | | | |
| 4 Тестирование | | | | | | | | | | | | | | 12.06.2024 | | | | |
| 5 Применение | | | | | | | | | | | | | | 17.06.2024 | | | | |
| Заключение. Список использованных источников. Приложения | | | | | | | | | | | | | | 21.06.2024 | | | | |
| **Графическая (практическая) часть курсового проекта (курсовой работы)** | | | | | | | | | | | | | | | | | | |
| Лист 1- | | Схема работы системы (А1) | | | | | | | | | | | | 19.06.2024 | | | | |
| Лист 2- | |  | | | | | | | | | | | | | | | | |
| Дата выдачи | | | 06 мая | | 20 | 24 | | г. | | | | | | | |
| Срок сдачи | | | 24 июня | | 20 | 24 | | г. | | | | | | | |
|  | | | | | |  | |  | |  | |  | | | | | |
| Руководитель  курсового проекта (курсовой работы) | | | | | |  |  | | | |  | | | |  | | | |
|  | | | | | |  | (подпись) | | | | | |  | | | | (инициалы, фамилия) | |
| Подпись учащегося | | | | | |  | | | | | |

**Содержание**

[Введение 4](#_Toc170198760)

[1 Анализ задачи 5](#_Toc170198761)

[1.2 Инструменты разработки 5](#_Toc170198762)

[1.3 Требования к приложению 6](#_Toc170198763)

[2. Проектирование задачи 8](#_Toc170198764)

[2.1 Организация данных 8](#_Toc170198765)

[2.2 Процессы 8](#_Toc170198766)

[2.3 Описание внешнего пользовательского интерфейса 8](#_Toc170198767)

[3 Реализация 10](#_Toc170198768)

[3.1 Структура программы 10](#_Toc170198769)

[3.1.1 Структура и описание процедур и функций пользователя 10](#_Toc170198770)

[3.1.2 Описание использованных компонентов 14](#_Toc170198771)

[3.2 Спецификация программы 14](#_Toc170198772)

[5 Руководство пользователя 18](#_Toc170198773)

[5.1 Общие сведения о программном продукте 18](#_Toc170198774)

[5.2 Инсталляция 18](#_Toc170198775)

[5.3 Выполнение программы 18](#_Toc170198776)

[5.3.1 Запуск программы 18](#_Toc170198777)

[5.3.3 Завершение работы с программой 23](#_Toc170198778)

[5.4 Использование системы справочной информации 24](#_Toc170198779)

[Заключение 25](#_Toc170198780)

Список использованных источников………………………………………………27

Приложение А Листинг программы ………………………………………………….

Изм.

Кол

Лист №док

Подпись

Дата

Лист

3

КП 2-40 01 01.33.41.11.24 ПЗ

Разраб.

Кузьмицкая

Пров.

Шатова

Н. контр.

Утв.

Разработка игрового приложения «Пиксель»

Стадия

Листов

УО ГГПК

# Введение

Разработка игрового приложения «Пиксель» представляет собой увлекательный процесс, который позволяет пользователям окунуться в мир пиксельной графики и насладиться необычным геймплеем. В современном мире все больше людей стремятся найти развлечения и отдых, которые соответствуют их интересам и предпочтениям. Поэтому создание игрового приложения с использованием пиксельной графики имеет высокую актуальность и востребованность.

Пояснительная записка к данной курсовой работе будет структурирована следующим образом.

В первом разделе «Анализ задачи» будет представлено исследование предметной области, обзор существующих решений и определение организационно-экономической сущности поставленной задачи.

В разделе «Проектирование задачи» будут рассмотрены основные аспекты создания программного продукта, включая организацию данных, пользовательский интерфейс и разработку системы справочной информации.

Далее в третьем разделе «Реализация задачи» будут подробно описаны элементы и объекты, используемые при создании игрового приложения «Пиксель». Важным этапом работы над проектом будет тестирование, которое будет подробно описано в четвертом разделе­­­ – «Тестирование». Заключительный раздел «Применение» будет представлять информацию о назначении приложения, его области применения и способах использования справочной системы.

В завершении пояснительной записки будет приведено заключение, в котором будут сформулированы основные результаты проделанной работы, в том числе использованные методы и средства разработки. Список использованных источников будет представлен в разделе «Литература», а в приложениях будут приведены листинг программы с комментариями и графическая схема работы системы.

# Анализ задачи

* 1. **Постановка задачи**

**Наименование задачи:** Разработка игрового приложения «Пиксель».

**Цель разработки:** игровое приложение будет разрабатываться исключительно в развлекательных целях, а также даст возможность окунутся в мир пиксельной графики.

**Назначение:** Приложение «Пиксель» предназначено для развлечения и отдыха, предоставляя пользователям инструмент для создания и взаимодействия с пиксельным искусством.

**Предметная область:** будет изучена предметная область на основе таких культовых игр, таких как Снайпер (Тир) и Arkanoid, разработана концепция, на основе которой, можно будет построить и реализовать модель.

**Периодичность использования:** зависит от нужд потребителя, может использоваться ежедневно.

**Источники и способы получения данных:** проект будет содержать 3 игры, которые будут иметь по 3 уровня сложности. Каждый уровень сложности будет иметь свои уникальные характеристики (фон, цветовая гамма, музыка).

Разрабатываемый программный продукт позволит выполнить следующие действия:

* дать возможность выбора игры;
* осуществить игровую сессию;
* возможность самостоятельно влиять на игровой процесс.

# Инструменты разработки

Для разработки данного проекта выбрана среда Delphi (Delphi 11), так как это среда объектно-ориентированного программирования, относящаяся к классу RAD – (Rapid Application Development – «Средство быстрой разработки приложений»), реализованная на Object Pascal. Используется для разработки визуализированного представления программного обеспечения.

Также для разработки программы необходимы:

* + - Google-браузер – нужен для нахождения информации и картинок персонажей игры, фоновых изображений;
    - Word 2021 – нужен для написания пояснительной записки;
    - Adobe Photoshop CC 2019 – нужен для создания и редактирования изображений;
    - Проигрыватель Windows Media – нужен для воспроизведения аудио файлов;
    - Smart Install Maker – нужен для создания инсталятора.

При разработке данного программного продукта был использован компьютер со следующими характеристиками:

* + - Процессор: AMD Ryzen 5 5500U with Radeon Graphics 2.10 GHz;
    - ОЗУ: 16Gb;
    - Память: HDD 222Gb;
    - ОС – Windows 10 Pro.

# Требования к приложению

На этапе изучения предметной области было выявлено несколько требований, предъявляемых к решаемой задаче. При разработке пользовательских форм следует учитывать следующие аспекты:

* простой и интуитивно понятный интерфейс;
* широкоформатные окна форм;
* привлекательный визуальный дизайн;
* наличие информации о правилах игры;
* удобные кнопки.

Примерные минимальные системные требования:

* процессор с тактовой частотой 1.8 ГГц;
* 1 ГБ оперативной памяти;
* не менее 500 МБ свободного места на жестком диске.

Рекомендуемые системные требования:

* двухъядерный процессор AMD Athlon или эквивалентный процессор Intel (или более мощный);
* 2 ГБ оперативной памяти;
* не менее 500 МБ свободного места на жестком диске.

При создании интерфейса приложения использован темный фон с яркими цветовыми акцентами. Основные разделы доступны с первого экрана. Все окна имеют четкую визуальную структуру элементов. Пользователю предоставлен дружелюбный интерфейс, где текст размещен удобным образом. Размеры объектов (персонажи, препятствия) выбраны с учетом легкости управления и игры.

Возможные ошибки, вызванные неправильным взаимодействием пользователя с графическим интерфейсом, не должны повлиять на функциональность программы. Программа должна быть надежной, без каких-либо сбоев и неполадок при работе.

# 2. Проектирование задачи

# 2.1 Организация данных

Основной способ хранения данных в приложении представлен bmp, png-графикой для сохранения текстур персонажей и фоновых изображений.

Аудиофайлы формата mp3 использованы для звукового оформления игрового процесса. Справочная информация доступна через файл справки, содержащий сведения о приложении и его правилах эксплуатации.

Основные функции приложения включают в себя:

* вхождение в игровую сессию;
* прохождение уровня для перехода к следующему;
* просмотр результата игры.

Путем выбора пункта меню «Cправка» пользователь имеет возможность ознакомиться с информацией о разработчике приложения.

## 2.2 Процессы

Согласно всем перечисленным требованиям и указаниям, которые были рассмотрены в разделе «Анализ задачи», было определено, чем конкретно должна заниматься разрабатываемая программа. Главной её задачей является прохождение игровой сессии.

Для реализации задач используются процедуры и функции. С помощью процедуры, например, будет осуществляться передвижение персонажа по игровому полю.

Персонажи и игровой уровень генерируются на случайной основе. На форме с игрой загружаются изображения фоновых текстур, целей.

Далее после того, как пользователь начнёт игру, он может пройти уровень игры для того, чтобы перейти к новой уровню, в случае же проигрыша переходит на таблицу результатов. После прохождения уровня, пользователю высветится сообщение о возможности перехода на следующий уровень.

# 2.3 Описание внешнего пользовательского интерфейса

Одним из важных аспектов при выполнении курсового проекта является настройка взаимодействия между пользователем и программой. Это напрямую зависит от подхода программиста к разработке программы, от выбора компонентов и методов автоматизации. Особое внимание следует уделить интерфейсу программы. Дизайн программы должен быть спроектирован таким образом, чтобы пользователь мог с легкостью понять, что от него требуется.

Важные функциональные элементы следует графически выделять, чтобы обеспечить визуализацию интерактивных возможностей приложения. В случае затруднений пользователя с восприятием интерфейса, следует предусмотреть возможность запроса подсказки.

## 3 Реализация

### 3.1 Структура программы

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

1. Модуль Main – отвечает за создание главного меню игры;
2. Модуль Arcanoid– отвечает за реализацию мини игры «Арканоид»;
3. Модуль Sniper­–отвечает за реализацию мини игры «Снайпер»;
4. Модуль Result– отвечает за создание таблицы результатов;
5. Модуль Zastavka – отображается экран загрузки;

### 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 FormCreate(Sender: TObject); | Main.pas | - | задает настройки формы при ее создании |
| procedure CreateParams(var Params: TCreateParams); override; | Main.pas | - | задает параметры создания окна |
| procedure Image1Click(Sender: TObject); | Main.pas | Image1 | скрывает текущей формы (Form1) и отображение следующей формы (Form2) |
| procedure Image2Click(Sender: TObject); | Main.pas | Image2 | Скрывает элементы (Image1, Image2, Image3, Image6) и возвращает остальные элементы (Memo1, Image5, Image6 ) |
| procedure Image3Click(Sender: TObject); | Main.pas | Image3 | скрывает текущей формы (Form1) и отображение следующей формы (Form3) |

Продолжение Таблицы 1

|  |  |  |  |
| --- | --- | --- | --- |
| procedure FormClose(Sender: TObject; var Action: TCloseAction); | Main.pas | - | Устанавливает действие закрытия формы как освобождение ресурсов |
| procedure Edit1KeyDown(Sender: TObject; var Key: Word; Shift: TShiftState); | Main.pas | Edit1 | обрабатывает нажатие клавиши в поле ввода. Если нажата клавиша «Enter», скрывает поле ввода и показывает элементы управления |
| procedure Image6Click(Sender: TObject); | Main.pas | Image6 | Скрывает элементы (Memo1, Image5, Image6 ) и возвращает остальные элементы (Image1, Image2, Image3, Image6) |
| procedure N1Click(Sender: TObject); | Main.pas | MenuItem | Запускает справочную справку |
| procedure N2Click(Sender: TObject); | Main.pas | MenuItem | Закрывает приложение |
| procedure Timer1Timer(Sender: TObject); | Zastavka.pas | Timer1 | обновляет позицию прогресс-бара в соответствии со значением и изменяет текст в Label на случайный текст из массива |
| procedure FormCreate(Sender: TObject); | Zastavka.pas | - | устанавливает начальные значения переменных и массива с текстами и случайным образом устанавливает их в Label1 |
| procedure FormCreate(Sender: TObject); | Arcanoid.pas | - | задает настройки формы при ее создании |
| procedure CreateParams(var Params: TCreateParams); override; | Arcanoid.pas | - | задает параметры создания окна |
| procedure Timer1Timer(Sender: TObject); | Arcanoid.pas | Timer1 | обновляет положение мяча, проверяет столкновения с кирпичами и платформой, а также условия завершения уровня |

Продолжение Таблицы 1

|  |  |  |  |
| --- | --- | --- | --- |
| procedure ImageMouseMove(Sender: TObject; Shift: TShiftState; X, Y: Integer); | Arcanoid.pas | Image | отслеживает движение мыши и передвигает платформу в соответствии с положением курсора |
| procedure fonClick(Sender: TObject); | Arcanoid.pas | Image fon | обрабатывает событие клика на объекте fon и позволяет выбирать фон игры |
| procedure Image3Click(Sender: TObject); | Arcanoid.pas | Image 3 | обрабатывает событие клика на объекте Image3 и запускает новый уровень игры |
| procedure musicClick(Sender: TObject); | Arcanoid.pas | Image music | обрабатывает событие клика на объекте music и позволяет выбирать музыкальное сопровождение игры |
| procedure Image4Click(Sender: TObject); | Arcanoid.pas | Image4 | обрабатывает событие клика на объекте Image4 и переходит к экрану результатов |
| procedure BackgroundMouseDown(Sender: TObject; Button: TMouseButton; Shift: TShiftState; X, Y: Integer); | Arcanoid.pas | Image Background | обрабатывает событие нажатия на фон игры и запускает игру |
| procedure SafeAr; | Arcanoid.pas | - | Сохраняет текущий результат игрока в файл рекордов |
| procedure RestartGame; | Arcanoid.pas | - | перезапуск игры со сбросом уровня, очков и размещением новых кирпичей на поле |
| procedure ResetBall; | Arcanoid.pas | - | возвращение мяча в начальное положение |
| procedure HideGameOverElements; | Arcanoid.pas | - | скрытие элементов интерфейса, связанных с завершением игры |
| constructor TBrick.Create(AOwner: TComponent); | Arcanoid.pas | Shape | создаёт кирпич с заданными параметрами (формой и цветом) |
| procedure TimeTimerTimer(Sender: TObject); | Sniper.pas | Timer | обновляет отображение времени в игре |
| procedure ShootTimerTimer(Sender: TObject); | Sniper.pas | Timer | управляет отображением и перемещением цели в игре |
| Продолжение Таблицы 1 |  |  |  |
| procedure BirdImageMouseDown(Sender: TObject; Button: TMouseButton; Shift: TShiftState; X, Y: Integer); | Sniper.pas | Image BirdImage | обрабатывает событие нажатия на цель (птицу) в игре, увеличивает количество очков игрока и проверяет условия окончания уровня |
| procedure FormCreate(Sender: TObject); | Sniper.pas | - | выполняет действия при создании формы игры |
| procedure Image3Click(Sender: TObject); | Sniper.pas | Image3 | обрабатывает событие клика на объекте Image3 и запускает новый уровень игры |
| procedure SleepTimer(Sender: TObject); | Sniper.pas | Timer | скрывает изображение цели после нажатия на неё |
| procedure fonClick(Sender: TObject); | Sniper.pas | Image fon | обрабатывает событие клика на объекте fon и позволяет выбирать фон игры |
| procedure musicClick(Sender: TObject); | Sniper.pas | Image music | обрабатывает событие клика на объекте music и позволяет выбирать музыкальное сопровождение игры |
| procedure Image4Click(Sender: TObject); | Sniper.pas | Image4 | обрабатывает событие клика на объекте Image4 и переходит к экрану результатов |
| procedure SafeSN; | Sniper.pas | - | предназначена для сохранения результатов игры в текстовый файл |
| procedure RestartGame; | Sniper.pas | - | инициализирует начало новой игры, сбрасывает уровень, количество очков и запускает таймеры для отслеживания времени и цели |
| procedure Image1Click(Sender: TObject); | Result.pas | Image1 | загружает результаты игры «Arcanoid» из файла «РЕкорды/Arcanoid.txt» и отображает их в ListBox1. |

Продолжение Таблицы 1

|  |  |  |  |
| --- | --- | --- | --- |
| procedure Image3Click(Sender: TObject); | Result.pas | Image3 | загружает результаты игры «Sniper» из файла «РЕкорды/Sniper.txt» и отображает их в ListBox1. |
| procedure Image6Click(Sender: TObject); | Result.pas | Image6 | скрывает текущую форму и открывает форму Main, при этом скрывая текстовое поле и активируя элементы управления на форме Main |
| procedure Image2Click(Sender: TObject); | Result.pas | Image2 | завершает работу приложения |

### 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 |
| TForm | Form1, Form2, Form3, Form4, Form5 | Используется для создания главного меню проекта |
| TImage | Form1, Form2, Form3, Form4, Form5 | Используется как кнопка |
| TTimer | Form2, Form3, Form4 | Управление временем и таймингом действий |
| TLabel | Form2, Form3, Form4 | Отображение надписей на форме |
| TPanel | Form2, Form3 | Группировка и размещение других компонентов |
| TMediaPlayer | Form2, Form3 | Используется для воспроизведения музыки |
| TOpenDialog | Form2, Form3 | Используется для открытия файлов |
| TEdit | Form1 | Используется для ввода текста |
| TListBox | Form5 | Используется для отображения текста |
| TShape | Form2 | Используется для создания фигур |
| TProgressBar | Form4 | индикатор прогресса загрузки |
| TMemo1 | Form1 | компонент для отображения помощи |
| TMainMenu1 | Form1 | главное меню с пунктами «Справка» и «Выход» |

### 3.2 Спецификация программы

Точное название проекта и его состав приводится в Таблице 3.

Таблица 3 – Спецификация программы

|  |  |
| --- | --- |
| Имя файла | Назначение |
| Pixel.exe | Исполняемый файл проекта |
| Help.chm | Файл справки |
| взрыв.png | картинка |
| NEXT.png | Картинка для кнопки «Далее» |
| Пиксель.bmp | картинка |
| помощь.bmp | Картинка для кнопки «Помощь» |
| Mmusic.bmp | Картинка для кнопки «Музыка» |
| Return.bmp | Картинка для кнопки «Вернуться» |
| background.bmp | Картинка для кнопки «Фон» |
| istockphoto-1180169333-1024x1024.png | картинка |
| Rules.bmp | картинка |
| Arcanoid.txt | Запись рекордов «Арканоид» |
| Sniper.txt | Запись рекордов «Снайпер» |
| Фон1.bmp | фон |
| Фон2.bmp | фон |
| Фон3.bmp | фон |
| Фон4.bmp | фон |
| Фон5.bmp | фон |
| Фон6.bmp | фон |
| Фон7.bmp | фон |
| Фон8.bmp | фон |
| Фон9.bmp | фон |
| Фон10.bmp | фон |
| Gilneas.mp3 | музыка |
| kim-lightyear-you-and-i-161104.mp3 | музыка |
| Mists of Pandaria.mp3 | музыка |
| Thomas\_Bergersen\_-\_Empire\_of\_Angels\_48079278.mp3 | музыка |
| Линия Танца - Гадкий Утенок.mp3 | музыка |
| Линия Танца - Зима.mp3 | музыка |
| Линия танца - Начало.mp3 | музыка |
| Линия танца - Равнины.mp3 | музыка |
| Линия Танца - Часы.mp3 | музыка |
| Линия Танца (OST) - Пустыня.mp3 | музыка |
| Arcanoid.dfm | форма Arcanoid |
| Arcanoid.pas | Файл программного модуля для формы Form2 |
| Main.dfm | форма Main |
| Main.pas | Файл программного модуля для формы Form1 |
| Pixel.dpr | Файл проекта, связывает все файлы из которых состоит приложение |
| Result.dfm | форма Result |
| Продолжение Таблицы 3 |  |
| Result.pas | Файл программного модуля для формы Form5 |
| Sniper.dfm | форма Sniper |
| Sniper.pas | Файл программного модуля для формы Form3 |
| Zastavka.dfm | форма Zastavka |
| Zastavka.pas | Файл программного модуля для формы Form4 |

4 Тестирование

В процессе разработки данного софта были устранены многие ошибки и недочеты на этапе реализации проекта. После завершения испытания реализации программного продукта произведено основательное функциональное тестирование. Функциональное тестирование необходимо для обеспечения независимой работы всех компонентов программы. Результаты тестирования представлены в отчете, представленном в таблице 4.

Таблица 4 – Отчёт результатах тестирования

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Тест | Ожидаемый результат | Физический результат | Результат  тестирования |
| 1 | Проверка входа пользователя в систему | Открытие главного окна | Открытие главного окна | Выполнено |
| 2 | Проверка кнопки  «Арканоид» | Открытие формы мини игры «Арканоид» | Открытие формы мини игры «Арканоид» | Выполнено |
| 3 | Проверка кнопки  «Снайпер» | Открытие формы мини игры «Снайпер» | Открытие формы мини игры «Снайпер» | Выполнено |
| 4 | Проверка кнопки  «Справка» | Открытие справочного файла | Открытие справочного файла | Выполнено |
| 5 | Проверка кнопки  «Посмотреть результат» | Открытие формы с результатами | Открытие формы с результатами | Выполнено |
| 6 | Проверка кнопки  «Далее» | Переход на следующий уровень | Переход на следующий уровень | Выполнено |
| 7 | Проверка кнопки «Фон» | Открытие файла для фона | Открытие файла для фона | Выполнено |
| 8 | Проверка кнопки  «Музыка» | Открытие медиафайла | Открытие медиафайла | Выполнено |
| 9 | Проверка кнопки  «Помощь» | Открытие правил игры | Открытие правил игры | Выполнено |
| 10 | Проверка кнопки  «Вернуться» | Возвращение на главную форму | Возвращение на главную форму | Выполнено |
| 11 | Проверка кнопки  «Выход» | Выход из программы | Выход из программы | Выполнено |

Программный продукт был тщательно протестирован, и в результате обнаружено множество исправлений, таких как исправление неполадок с переходом между изображениями по нажатию стрелки и правильное управление доступом к квестам в зависимости от их прохождения. Все элементы программы были проверены и подтверждено, что они корректно выполняют свои функции в соответствии с установленными процедурами.

# 5 Руководство пользователя

# 5.1 Общие сведения о программном продукте

Программа «Пиксель» – это уникальная игра, созданная для тех, кто испытывает ностальгию по классическим пиксельным играм прошлого, таким как «Арканоид» и «Снайпер (Тир)». Она объединяет лучшие черты этих легендарных игр с современными элементами игрового дизайна, предлагая как знакомый, так и новый игровой опыт.

Для комфортного запуска и игры в «Пиксель» требуются следующие минимальные системные характеристики:

* Операционная система: Windows 7 и выше, macOS 10.12 и выше;
* Процессор: 1.6 GHz или выше;
* Оперативная память: 2 GB RAM;
* Графика: DirectX 9.0c совместимая видеокарта с 256 MB видеопамяти;
* Жесткий диск: 500 MB свободного места;
* Дополнительно: Звуковая карта, клавиатура и мышь.

# 5.2 Инсталляция

Для того, чтобы установить программу необходимо запустить файл

Установщик.exe. Появится окно установки приложения «Pixel»

Затем достаточно следовать приведенной инструкции установки приложения.

Примечание: перед установкой на свободном диске создайте папку.

# 5.3 Выполнение программы

# 5.3.1 Запуск программы

Данную программу можно запустить различными способами.

Первым из них является запуск с помощью ярлыка на рабочем столе.

Необходимо дважды щелкнуть левой кнопкой мыши на ярлыке с названием «Пиксель».

Вторым способом является запуск из каталога, в который устанавливалось

приложение (по умолчанию C:\Program Files (x86)\Кузьмицкая Анжелика\Пиксель).

**5.3.2 Инструкции по работе с программой**

После запуска приложения на экране нас встречает заставка на рисунок 1.

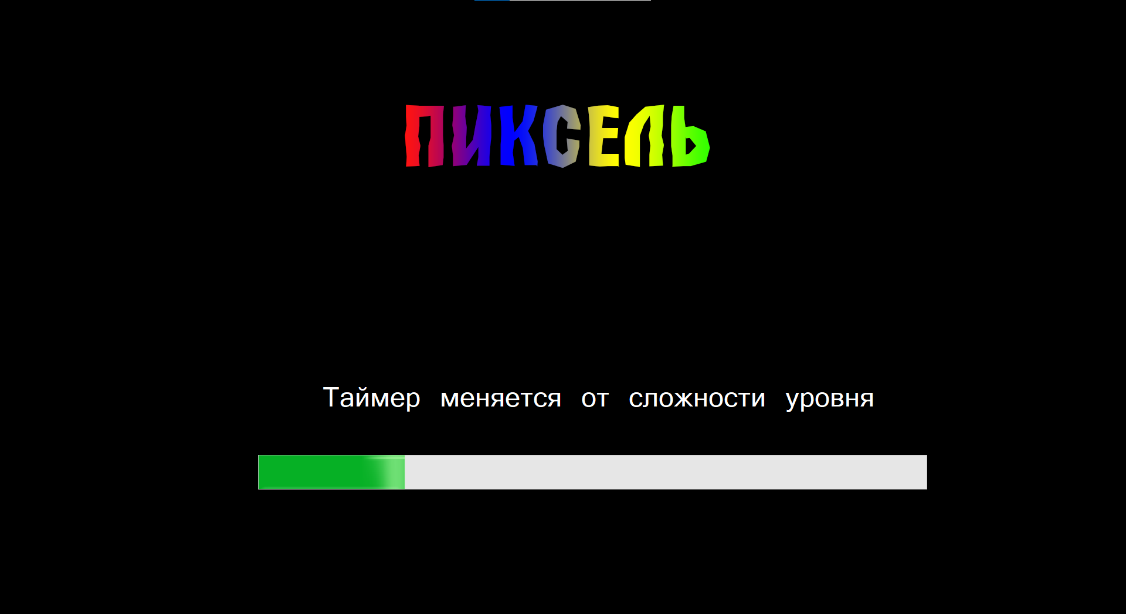
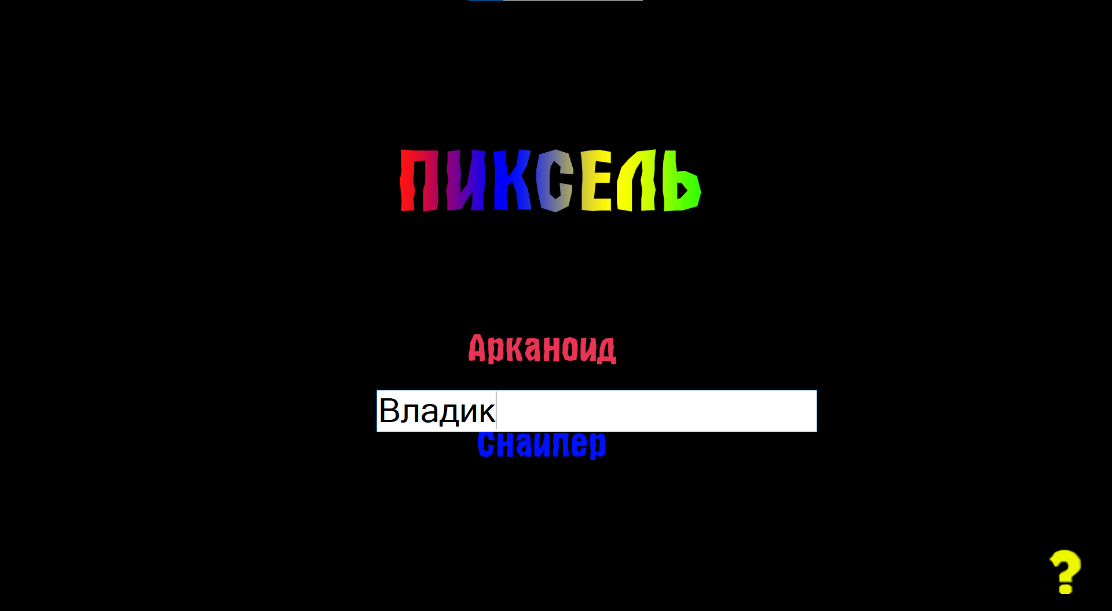


Рисунок 1 – заставка

Дальше вы увидите главное меню, где вам потребуется ввести свое имя (рисунок 2).

Рисунок 2 - имя игрока

После ввода имени будут доступны две игры-«Арканоид» и «Снайпер» (рисунок 3).



Рисунок 3 - Главное меню

Также будет рядом с выбором игры находится кнопка «Помощь». «Помощь» подскажет вам как играть в игры, которые были представлены в самом начале (рисунок 4).

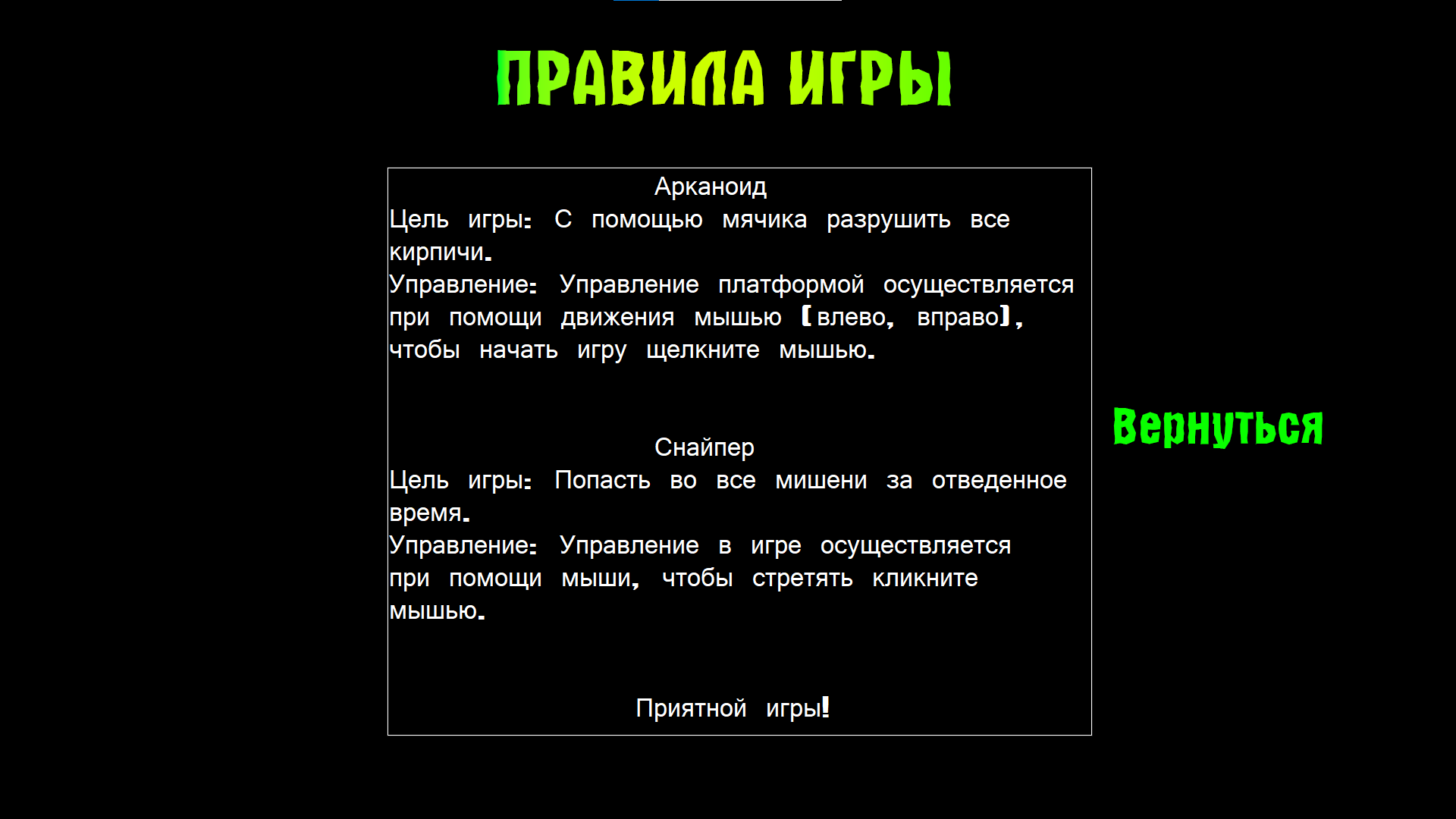


Рисунок 4 - Правила игры

После того, как вы ознакометесь с правилами игры вы можете вернуться на главное меню. После выбора игры вы сразу же ее начнете. Как только вы выбирите игру «Арканоид», то вам потребуется щелкнуть мышью, чтобы мяч начал движение (рисунок 5).



Рисунок 5- «Арканоид»

Целью игры же является разрушить все кирпичи так, чтобы мяч не упал. Если же вы не разрушили все кирпичи, то у вас появится надпись «Конец игры» и кнопка «Посмотреть результаты», которая приведет вас к таблице рекордов (рисунок 6).

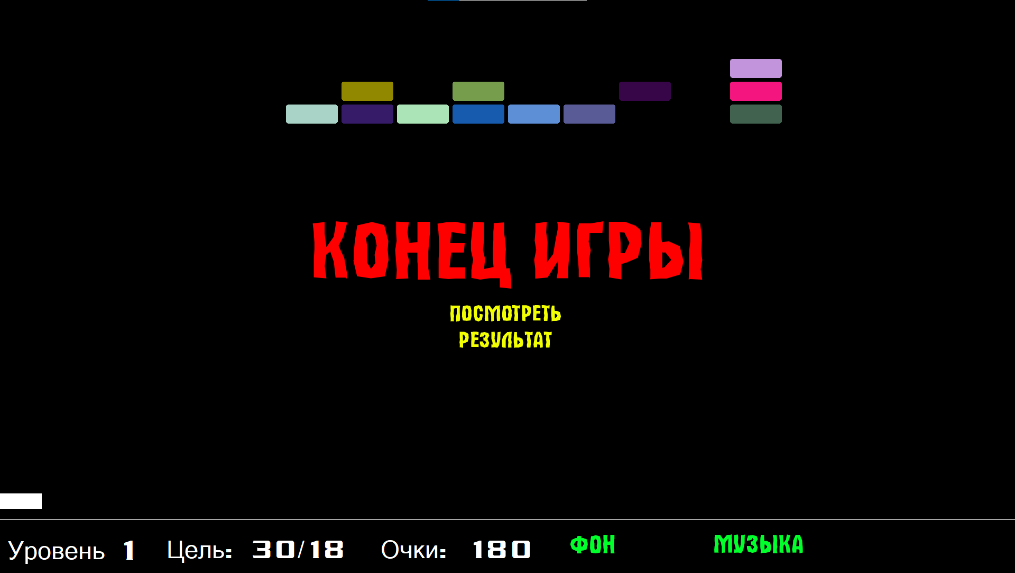


Рисунок 6 - Проиграл

Если же вы разрушили все кирпичи, то вы можете перейти на следующий уровень или же перейти к таблице рекордов, если вам наскучило играть (рисунок 7).

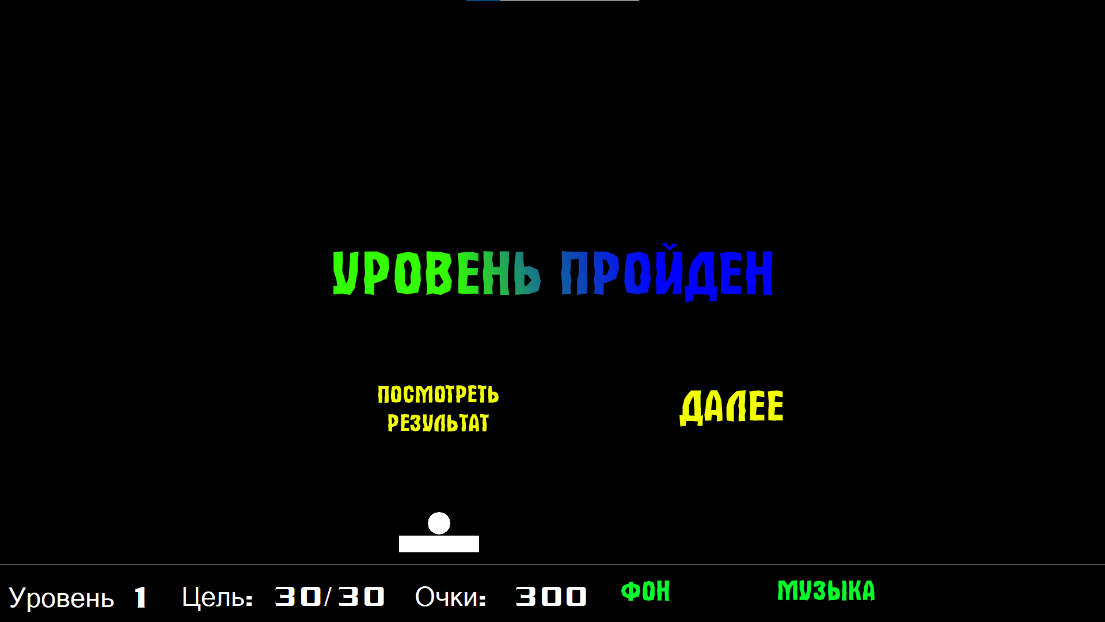


Рисунок 7 - Выйграл

При выборе игры «Снайпер» вашей целью будет успеть за некоторый промежуток времени подстрелить определенное количество мишеней (рисунок 8).

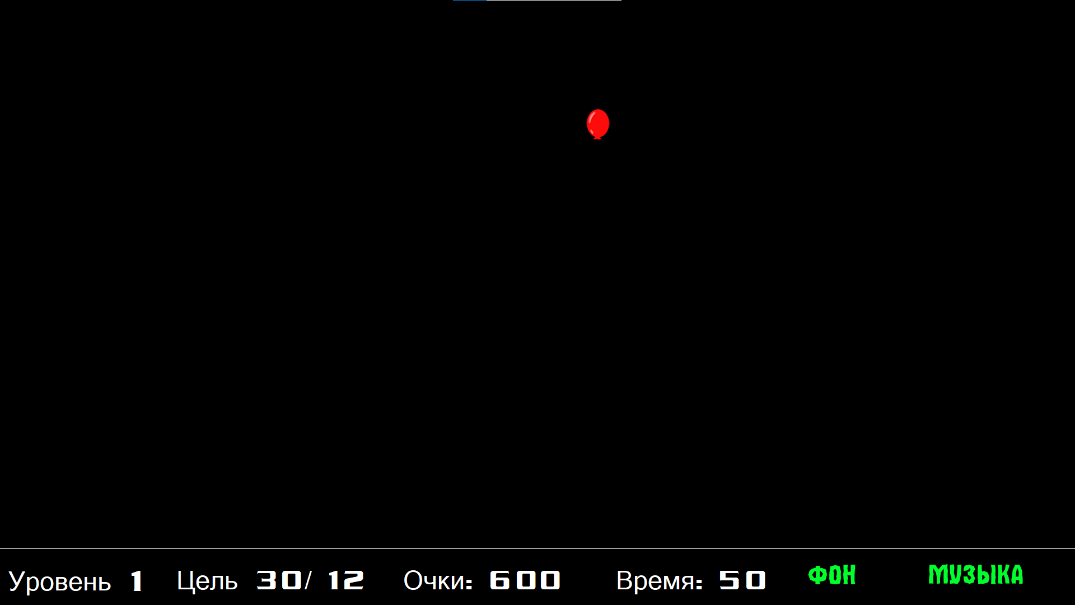


Рисунок 8 – «Снайпер»

Если цель провалилась вы пережодете к таблице результатов, если все получилось-переход на следующий уровень. Также во время игр вы можете поменять фон и музыку, нажав на соответствующие кнопки (рисунок 9).



Рисунок 9 - Изменение фона и музыки

Находясь на таблице рекордов вы можете посмотреть на результать своей игры кликнув по ее названию. вы также можете перейти главную форму, чтобы сыграть в другую игру (рисунок 10). Также вы можете выйти с приложения нажав на кнопку «Выйти».

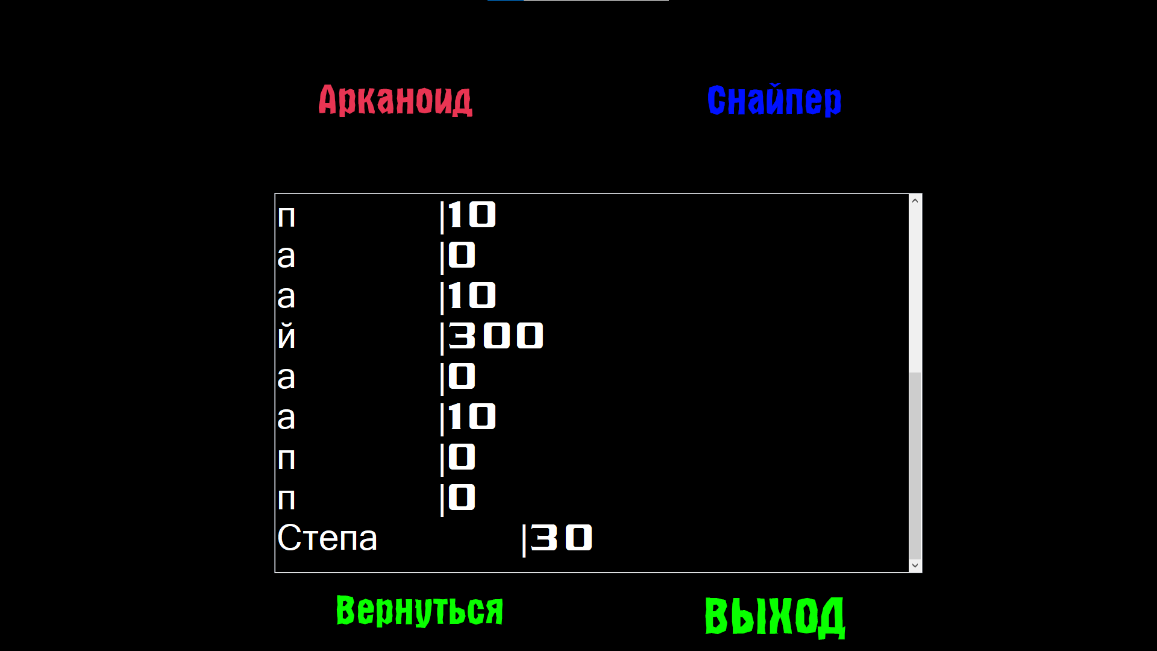


Рисунок 10 - Результаты

# 5.3.3 Завершение работы с программой

Завершить работу с программой можно несколькими способами:

* можно закрыть игровое приложение нажав пункт Файл – Выход;
* нажать на кнопку в таблице рекордов «Выход». (Рисунок 10).

# 5.4 Использование системы справочной информации

Справочную систему можно запустить с помощью пункта меню Справка (Рисунок 11).

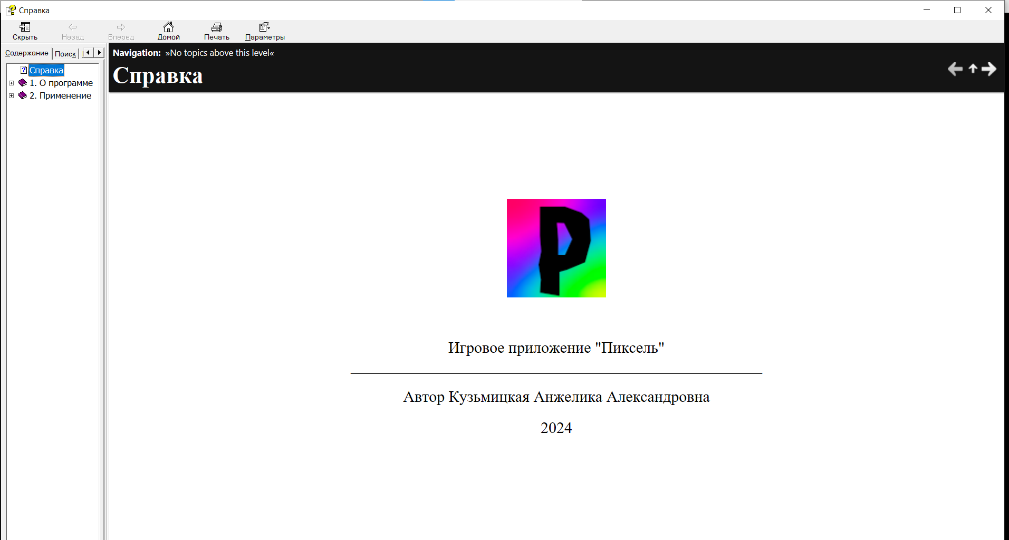


Рисунок 11 – Справочник

# Заключение

Разработка программного приложения «Пиксель» оказалась увлекательным и значимым проектом, который успешно соединил современные технологии и ностальгию по пиксельным играм прошлых десятилетий. Погружаясь в создание этого приложения, мы стремились не только возродить очарование пиксельной графики, но и предложить пользователям инновационные и полезные инструменты.

Идея создания «Пикселя» родилась из любви к классическим видеоиграм, которые подарили многим из нас незабываемые моменты детства. Пиксельная графика, несмотря на свою простоту, обладает уникальной эстетикой и способна вызывать теплые чувства. Наша цель заключалась в том, чтобы возродить эту магию и адаптировать её под современные реалии.

Процесс разработки включал несколько ключевых этапов:

* Анализ требований: Изучение потребностей и предпочтений пользователей, заинтересованных в пиксельной графике, позволило нам точно определить функциональность приложения;
* Проектирование: Разработана архитектура приложения, обеспечивающая гибкость и расширяемость. Особое внимание уделено интуитивно понятному интерфейсу с удобством использования;
* Разработка: Используя современные технологии и языки программирования, было создано надежное и высокопроизводительное приложение;
* Тестирование: Проведено комплексное тестирование для обеспечения стабильности и исправления всех выявленных ошибок.

Проект «Пиксель» стал воплощением нашей страсти к играм прошлого и стремления к инновациям. Мы верим, что приложение найдет отклик у широкой аудитории, пробуждая ностальгию и вдохновляя на творчество.

# Список использованных источников

1. cyberforum [Электронный ресурс]. – Электронные данные. – Режим доступа: https://www.cyberforum.ru. – Дата доступа: 30.05.2024

2. programmersforum [Электронный ресурс]. – Электронные данные. – Режим доступа: <https://delphisources.ru>. – Дата доступа: 02.06.2024

3. delphisources [Электронный ресурс]. – Электронные данные. – Режим доступа: https://delphi-manual.ru/. – Дата доступа: 09.06.2024

4. stackoverflow [Электронный ресурс]. – Электронные данные. – Режим доступа: https://ru.stackoverflow.com.– Дата доступа: 09.06.2024

Приложение А

unit Zastavka;

interface

uses

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

Vcl.Graphics, Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.ExtCtrls, Vcl.ComCtrls,

Vcl.StdCtrls;

type

TForm4 = class(TForm)

Image4: TImage;

Timer1: TTimer;

ProgressBar1: TProgressBar;

Label1: TLabel;

procedure Timer1Timer(Sender: TObject);

procedure FormCreate(Sender: TObject);

private

{ Private declarations }

ProgressValue: Double;

TargetValue: Integer;

RandomTexts: array of string;

public

{ Public declarations }

end;

var

Form4: TForm4;

implementation

{$R \*.dfm}

procedure TForm4.FormCreate(Sender: TObject);

begin

ProgressValue := 0;

ProgressBar1.Position := 0;

TargetValue := 100;

SetLength(RandomTexts, 4);

RandomTexts[0] := 'Загружаем';

RandomTexts[1] := 'Таймер меняется от сложности уровня';

RandomTexts[2] := 'Может сыграем?';

RandomTexts[3] := 'Расслабимся с "Пиксель"';

Label1.Caption := RandomTexts[Random(Length(RandomTexts))];

Label1.Alignment := taCenter;

end;

procedure TForm4.Timer1Timer(Sender: TObject);

begin

if ProgressValue < TargetValue then

begin

ProgressValue := ProgressValue + 10;

ProgressBar1.Position := Round(ProgressValue);

Label1.Caption := RandomTexts[Random(Length(RandomTexts))];

end

else

begin

Timer1.Enabled := False;

Close;

end;

end;

end.

unit Main;

interface

uses

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

Dialogs, StdCtrls, ExtCtrls, Arcanoid, Sniper, ShellAPI, Vcl.Menus;

type

TForm1 = class(TForm)

Image1: TImage;

Edit1: TEdit;

Image2: TImage;

Image3: TImage;

Image4: TImage;

Image5: TImage;

Memo1: TMemo;

Image6: TImage;

MainMenu1: TMainMenu;

N1: TMenuItem;

N2: TMenuItem;

N3: TMenuItem;

procedure FormCreate(Sender: TObject);

procedure CreateParams(var Params: TCreateParams); override;

procedure Edit1KeyDown(Sender: TObject; var Key: Word; Shift: TShiftState);

procedure Image1Click(Sender: TObject);

procedure Image3Click(Sender: TObject);

procedure Image2Click(Sender: TObject);

procedure Image6Click(Sender: TObject);

procedure FormClose(Sender: TObject; var Action: TCloseAction);

procedure N1Click(Sender: TObject);

procedure N3Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

procedure ToggleControls(Enable: Boolean);

end;

var

Form1: TForm1;

implementation

{$R \*.dfm}

uses Result;

procedure TForm1.FormClose(Sender: TObject; var Action: TCloseAction);

begin

Action := caFree;

end;

procedure TForm1.CreateParams(var Params: TCreateParams);

begin

inherited CreateParams(Params);

end;

procedure TForm1.Edit1KeyDown(Sender: TObject; var Key: Word; Shift: TShiftState);

begin

if Key = VK\_RETURN then

begin

if Edit1.Text <> '' then

begin

Edit1.Hide;

ToggleControls(True);

end;

end;

end;

procedure TForm1.FormCreate(Sender: TObject);

begin

Form1.Hide;

BorderStyle := bsNone;

Top := 0;

Left := 0;

Width := Screen.Width;

Height := Screen.Height;

ToggleControls(False);

end;

procedure TForm1.Image1Click(Sender: TObject);

begin

if Edit1.Text <> '' then

begin

Form2.PlayerName := Edit1.Text;

Form1.Hide;

Form2.Show;

end;

end;

procedure TForm1.Image2Click(Sender: TObject);

begin

Image1.Hide;

Edit1.Hide;

Image2.Hide;

Image3.Hide;

Image4.Hide;

Image5.Visible := True;

Image6.Visible := True;

Memo1.Visible := True;

end;

procedure TForm1.Image3Click(Sender: TObject);

begin

if Edit1.Text <> '' then

begin

Form3.PlayerName := Edit1.Text;

Form1.Hide;

Form3.Show;

end;

end;

procedure TForm1.Image6Click(Sender: TObject);

begin

Image5.Hide;

Image6.Hide;

Memo1.Hide;

Image1.Visible := True;

Image2.Visible := True;

Image3.Visible := True;

Image4.Visible := True;

end;

procedure TForm1.N1Click(Sender: TObject);

begin

ShellExecute(0, PChar('Open'), PChar('Help.chm'), nil, nil, SW\_SHOW);

end;

procedure TForm1.N3Click(Sender: TObject);

begin

Close;

end;

procedure TForm1.ToggleControls(Enable: Boolean);

begin

Image1.Enabled := Enable;

Image2.Enabled := Enable;

Image3.Enabled := Enable;

end;

end.

unit Arcanoid;

interface

uses

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

Dialogs, StdCtrls, ExtCtrls, Types, Math, Vcl.Menus, Vcl.Imaging.jpeg,

Vcl.ExtDlgs, Vcl.MPlayer, Vcl.Imaging.pngimage;

type

TBrick = class(TShape)

public

constructor Create(AOwner: TComponent); override;

end;

TForm2 = class(TForm)

OpenDialog1: TOpenDialog;

Timer1: TTimer;

Ball: TShape;

Paddle: TShape;

Panel1: TPanel;

Label1: TLabel;

Label2: TLabel;

Label3: TLabel;

Label4: TLabel;

Label5: TLabel;

Image1: TImage;

Image2: TImage;

Image3: TImage;

Image4: TImage;

fon: TImage;

music: TImage;

Background: TImage;

MediaPlayer1: TMediaPlayer;

Image6: TImage;

procedure FormCreate(Sender: TObject);

procedure CreateParams(var Params: TCreateParams); override;

procedure Timer1Timer(Sender: TObject);

procedure ImageMouseMove(Sender: TObject; Shift: TShiftState; X, Y: Integer);

procedure Image3Click(Sender: TObject);

procedure fonClick(Sender: TObject);

procedure musicClick(Sender: TObject);

procedure Image4Click(Sender: TObject);

procedure BackgroundMouseDown(Sender: TObject; Button: TMouseButton; Shift: TShiftState; X, Y: Integer);

procedure SafeAr;

procedure RestartGame;

private

{ Private declarations }

procedure ResetBall;

procedure HideGameOverElements;

public

{ Public declarations }

RecordFileName: string;

RecordFile: TextFile;

PlayerName: string;

GameStarted: Boolean;

end;

var

Form2: TForm2;

dx, dy: Integer;

Bricks: array of TBrick;

n: Integer;

Level: Integer;

implementation

{$R \*.dfm}

uses

Main, Result;

procedure TForm2.SafeAr;

begin

RecordFileName := ExtractFilePath(Application.ExeName) + 'РЕкорды/Arcanoid.txt';

AssignFile(RecordFile, RecordFileName);

if FileExists(RecordFileName) then

Append(RecordFile)

else

Rewrite(RecordFile);

Writeln(RecordFile, PlayerName + ' |' + Label5.Caption);

CloseFile(RecordFile);

end;

constructor TBrick.Create(AOwner: TComponent);

begin

inherited Create(AOwner);

Shape := stRoundRect;

Width := 80;

Height := 30;

Brush.Color := RGB(Random(255), Random(255), Random(255));

end;

procedure TForm2.CreateParams(var Params: TCreateParams);

begin

inherited CreateParams(Params);

Params.ExStyle := Params.ExStyle or WS\_EX\_TOPMOST;

end;

procedure TForm2.FormCreate(Sender: TObject);

var

i: Integer;

begin

Level := 1;

n := 30;

Top := 0;

Left := 0;

Width := GetSystemMetrics(SM\_CXSCREEN);

Height := GetSystemMetrics(SM\_CYSCREEN);

dx := 5;

dy := 5;

Paddle.Top := ClientHeight - Paddle.Height - 120;

Ball.Top := Paddle.Top - Ball.Height;

Ball.Left := Paddle.Left + (Paddle.Width - Ball.Width) div 2;

GameStarted := False;

SetLength(Bricks, n);

for i := 0 to n - 1 do

begin

Bricks[i] := TBrick.Create(Self);

Bricks[i].Parent := Self;

Bricks[i].Top := ClientHeight div 2 - (Bricks[i].Height + 5) \* 10 + (i div 10) \* (Bricks[i].Height + 5);

Bricks[i].Left := ClientWidth div 2 - (Bricks[i].Width + 5) \* 5 + (i mod 10) \* (Bricks[i].Width + 5);

end;

DoubleBuffered := True;

end;

procedure TForm2.ImageMouseMove(Sender: TObject; Shift: TShiftState; X, Y: Integer);

begin

Paddle.Left := X - Paddle.Width div 2;

if not GameStarted then

begin

Ball.Left := Paddle.Left + (Paddle.Width - Ball.Width) div 2;

end;

end;

procedure TForm2.Image3Click(Sender: TObject);

var

i: Integer;

begin

if Level = 1 then

begin

n := 50;

ResetBall;

Timer1.Interval := 10;

Label1.Caption := 'Уровень 2';

Label2.Caption := 'Цель 50/';

Label3.Caption := '0';

Image1.Visible := False;

Image3.Visible := False;

Image4.Visible := False;

SetLength(Bricks, n);

for i := 0 to n - 1 do

begin

Bricks[i] := TBrick.Create(Self);

Bricks[i].Parent := Self;

Bricks[i].Top := ClientHeight div 2 - (Bricks[i].Height + 5) \* 12 + (i div 5) \* (Bricks[i].Height + 35);

Bricks[i].Left := ClientWidth div 2 - (Bricks[i].Width + 5) \* 8 + (i mod 10) \* (Bricks[i].Width + 55);

end;

Inc(Level);

HideGameOverElements;

end

else if Level = 2 then

begin

n := 70;

ResetBall;

Timer1.Interval := 1;

Label1.Caption := 'Уровень 3';

Label2.Caption := 'Цель 70/';

Label3.Caption := '0';

Image1.Visible := False;

Image3.Visible := False;

Image4.Visible := False;

SetLength(Bricks, n);

for i := 0 to n - 1 do

begin

Bricks[i] := TBrick.Create(Self);

Bricks[i].Parent := Self;

Bricks[i].Top := ClientHeight div 2 - (Bricks[i].Height + 5) \* 10 + (i div 10) \* (Bricks[i].Height + 35);

Bricks[i].Left := ClientWidth div 2 - (Bricks[i].Width - 25) \* 10 + (i mod 10) \* (Bricks[i].Width + 1);

end;

HideGameOverElements;

end;

end;

procedure TForm2.Image4Click(Sender: TObject);

begin

SafeAr;

Form2.Hide;

Form5.Show;

HideGameOverElements;

RestartGame;

end;

procedure TForm2.BackgroundMouseDown(Sender: TObject; Button: TMouseButton; Shift: TShiftState; X, Y: Integer);

begin

if not GameStarted then

begin

GameStarted := True;

Timer1.Enabled := True;

HideGameOverElements;

end;

end;

procedure TForm2.Timer1Timer(Sender: TObject);

var

i, totalBricks, score: Integer;

allBricksDestroyed, ballHitBottom: Boolean;

begin

if not GameStarted then Exit;

Ball.Left := Ball.Left + dx;

Ball.Top := Ball.Top + dy;

totalBricks := 0;

score := StrToInt(Label5.Caption);

allBricksDestroyed := True;

ballHitBottom := False;

for i := 0 to n - 1 do

begin

if Ball.BoundsRect.IntersectsWith(Bricks[i].BoundsRect) and Bricks[i].Visible then

begin

dy := -dy;

Bricks[i].Visible := False;

Inc(totalBricks);

Inc(score, 10);

end;

if Bricks[i].Visible then

allBricksDestroyed := False;

end;

Label3.Caption := IntToStr(StrToInt(Label3.Caption) + totalBricks);

Label5.Caption := IntToStr(score);

if allBricksDestroyed then

begin

Image1.Visible := True;

Image3.Visible := True;

Image4.Visible := True;

Timer1.Enabled := False;

GameStarted := False;

ResetBall;

end;

if Ball.Left <= 0 then

dx := Abs(dx);

if Ball.Left + Ball.Width >= ClientWidth then

dx := -Abs(dx);

if Ball.Top <= 0 then

dy := Abs(dy);

if Ball.BoundsRect.IntersectsWith(Paddle.BoundsRect) then

begin

dy := -Abs(dy);

Image2.Visible := False;

end

else if Ball.Top + Ball.Height >= ClientHeight then

begin

ballHitBottom := True;

Image4.Visible := False;

end;

if ballHitBottom then

begin

Image2.Visible := True;

Image4.Visible := True;

Image4.Top := 565;

Image4.Left := 810;

Timer1.Enabled := False;

GameStarted := False;

end;

end;

procedure TForm2.fonClick(Sender: TObject);

begin

Timer1.Enabled := False;

if OpenDialog1.Execute then

begin

Timer1.Enabled := True;

Background.Picture.LoadFromFile(OpenDialog1.FileName);

end;

end;

procedure TForm2.musicClick(Sender: TObject);

begin

Timer1.Enabled := False;

if OpenDialog1.Execute then

begin

Timer1.Enabled := True;

MediaPlayer1.FileName := OpenDialog1.FileName;

MediaPlayer1.Open;

MediaPlayer1.Play;

end;

end;

procedure TForm2.ResetBall;

begin

Ball.Top := Paddle.Top - Ball.Height;

Ball.Left := Paddle.Left + (Paddle.Width - Ball.Width) div 2;

end;

procedure TForm2.HideGameOverElements;

begin

Image1.Visible := False;

Image2.Visible := False;

Image3.Visible := False;

Image4.Visible := False;

end;

procedure TForm2.RestartGame;

var

i: Integer;

begin

Level := 1;

n := 30;

Paddle.Top := ClientHeight - Paddle.Height - 120;

ResetBall;

GameStarted := False;

Label1.Caption := 'Уровень 1';

Label2.Caption := 'Цель 30/';

Label3.Caption := '0';

Label5.Caption := '0';

SetLength(Bricks, n);

for i := 0 to n - 1 do

begin

Bricks[i] := TBrick.Create(Self);

Bricks[i].Parent := Self;

Bricks[i].Top := ClientHeight div 2 - (Bricks[i].Height + 5) \* 10 + (i div 10) \* (Bricks[i].Height + 5);

Bricks[i].Left := ClientWidth div 2 - (Bricks[i].Width + 5) \* 5 + (i mod 10) \* (Bricks[i].Width + 5);

end;

HideGameOverElements;

Timer1.Enabled := False;

end;

end.

unit Sniper;

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants,

System.Classes, Vcl.Graphics, Vcl.Controls, Vcl.Forms, Vcl.Dialogs,

Vcl.ExtCtrls, Vcl.StdCtrls, Vcl.Imaging.pngimage, Vcl.MPlayer;

type

TForm3 = class(TForm)

Panel1: TPanel;

Label1: TLabel;

Label2: TLabel;

Label3: TLabel;

Label4: TLabel;

Label5: TLabel;

Label6: TLabel;

Label7: TLabel;

TimeTimer: TTimer;

ShootTimer: TTimer;

BirdImage: TImage;

Image1: TImage;

Image2: TImage;

Image3: TImage;

Image4: TImage;

Image6: TImage;

Sleep: TTimer;

fon: TImage;

music: TImage;

Background: TImage;

MediaPlayer1: TMediaPlayer;

OpenDialog1: TOpenDialog;

procedure TimeTimerTimer(Sender: TObject);

procedure ShootTimerTimer(Sender: TObject);

procedure BirdImageMouseDown(Sender: TObject; Button: TMouseButton; Shift: TShiftState; X, Y: Integer);

procedure FormCreate(Sender: TObject);

procedure Image3Click(Sender: TObject);

procedure SleepTimer(Sender: TObject);

procedure fonClick(Sender: TObject);

procedure musicClick(Sender: TObject);

procedure Image4Click(Sender: TObject);

procedure SafeSN;

procedure RestartGame;

private

t: Integer;

fx, fy: Integer;

n, p: Integer;

RunGame: Boolean;

TimeForShoot: Integer;

Level: Integer;

public

RecordFileName: string;

RecordFile: TextFile;

PlayerName: string;

end;

var

Form3: TForm3;

implementation

{$R \*.dfm}

uses

Result, Main;

procedure TForm3.SafeSN;

begin

RecordFileName := ExtractFilePath(Application.ExeName) + 'РЕкорды/Sniper.txt';

AssignFile(RecordFile, RecordFileName);

if FileExists(RecordFileName) then

Append(RecordFile)

else

Rewrite(RecordFile);

Writeln(RecordFile, PlayerName + ' |' + Label5.Caption);

CloseFile(RecordFile);

end;

procedure TForm3.fonClick(Sender: TObject);

begin

TimeTimer.Enabled := False;

ShootTimer.Enabled := False;

if OpenDialog1.Execute then

begin

Background.Picture.LoadFromFile(OpenDialog1.FileName);

TimeTimer.Enabled := True;

ShootTimer.Enabled := True;

end;

end;

procedure TForm3.FormCreate(Sender: TObject);

begin

RestartGame;

end;

procedure TForm3.Image3Click(Sender: TObject);

begin

if Level = 1 then

begin

n := 50;

p := 0;

t := 90;

TimeForShoot := 850;

RunGame := True;

TimeTimer.Enabled := True;

ShootTimer.Enabled := True;

Label1.Caption := 'Уровень 2';

Label2.Caption := 'Цель 50/';

Label3.Caption := '0';

Image1.Visible := False;

Image3.Visible := False;

Image4.Visible := False;

Inc(Level);

end

else if Level = 2 then

begin

n := 70;

p := 0;

t := 60;

TimeForShoot := 700;

Label1.Caption := 'Уровень 3';

Label2.Caption := 'Цель 70/';

Label3.Caption := '0';

RunGame := True;

TimeTimer.Enabled := True;

ShootTimer.Enabled := True;

Image1.Visible := False;

Image3.Visible := False;

Image4.Visible := False;

Inc(Level);

end;

end;

procedure TForm3.Image4Click(Sender: TObject);

begin

SafeSN;

Hide;

Form5.Show;

RestartGame;

end;

procedure TForm3.musicClick(Sender: TObject);

begin

TimeTimer.Enabled := False;

ShootTimer.Enabled := False;

if OpenDialog1.Execute then

begin

MediaPlayer1.FileName := OpenDialog1.FileName;

MediaPlayer1.Open;

MediaPlayer1.Play;

TimeTimer.Enabled := True;

ShootTimer.Enabled := True;

end;

end;

procedure TForm3.TimeTimerTimer(Sender: TObject);

begin

if (t > 0) and RunGame then

begin

t := t - 1;

Label7.Caption := IntToStr(t);

if t = 0 then

begin

RunGame := False;

TimeTimer.Enabled := False;

ShootTimer.Enabled := False;

Image2.Visible := True;

Image4.Visible := True;

end;

end;

end;

procedure TForm3.ShootTimerTimer(Sender: TObject);

begin

if RunGame then

begin

ShootTimer.Interval := TimeForShoot;

fx := Random(1400);

fy := Random(600);

BirdImage.Left := fx;

BirdImage.Top := fy;

end;

end;

procedure TForm3.SleepTimer(Sender: TObject);

begin

Image6.Visible := False;

Sleep.Enabled := False;

end;

procedure TForm3.BirdImageMouseDown(Sender: TObject; Button: TMouseButton; Shift: TShiftState; X, Y: Integer);

begin

if RunGame then

begin

p := p + 1;

Label3.Caption := IntToStr(p);

Label5.Caption := IntToStr(StrToInt(Label5.Caption) + 50);

Image6.Left := BirdImage.Left;

Image6.Top := BirdImage.Top;

Image6.Visible := True;

Sleep.Enabled := True;

if (Level = 1) and (p = 30) then

begin

Image1.Visible := True;

Image4.Visible := True;

Image3.Visible := True;

TimeTimer.Enabled := False;

ShootTimer.Enabled := False;

RunGame := False;

end

else if (Level = 2) and (p = 50) then

begin

Image1.Visible := True;

Image4.Visible := True;

Image3.Visible := True;

TimeTimer.Enabled := False;

ShootTimer.Enabled := False;

RunGame := False;

end

else if (Level = 3) and (p = 70) then

begin

Image1.Visible := True;

Image4.Visible := True;

Image3.Visible := False;

TimeTimer.Enabled := False;

ShootTimer.Enabled := False;

RunGame := False;

end;

end;

end;

procedure TForm3.RestartGame;

begin

Level := 1;

t := 100;

n := 0;

p := 0;

RunGame := True;

TimeForShoot := 1000;

Label1.Caption := 'Уровень 1';

Label2.Caption := 'Цель 30/';

Label3.Caption := '0';

Label5.Caption := '0';

Label7.Caption := IntToStr(t);

Image1.Visible := False;

Image2.Visible := False;

Image3.Visible := False;

Image4.Visible := False;

TimeTimer.Enabled := True;

ShootTimer.Enabled := True;

end;

end.

unit Result;

interface

uses

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

Vcl.Graphics, Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.StdCtrls, Vcl.ExtCtrls, System.IOUtils,

Vcl.Imaging.pngimage;

type

TForm5 = class(TForm)

Image1: TImage;

Image3: TImage;

ListBox1: TListBox;

Image6: TImage;

Image2: TImage;

procedure Image1Click(Sender: TObject);

procedure Image3Click(Sender: TObject);

procedure Image6Click(Sender: TObject);

procedure Image2Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form5: TForm5;

implementation

{$R \*.dfm}

uses Main, Sniper;

procedure TForm5.Image1Click(Sender: TObject);

var

Lines: TStringList;

begin

Lines := TStringList.Create;

try

if TFile.Exists('РЕкорды/Arcanoid.txt') then

begin

Lines.LoadFromFile('РЕкорды/Arcanoid.txt');

ListBox1.Items.Assign(Lines);

end;

finally

Lines.Free;

end;

end;

procedure TForm5.Image2Click(Sender: TObject);

begin

Application.Terminate;

end;

procedure TForm5.Image3Click(Sender: TObject);

var

Lines1: TStringList;

begin

Lines1 := TStringList.Create;

try

if TFile.Exists('РЕкорды/Sniper.txt') then

begin

Lines1.LoadFromFile('РЕкорды/Sniper.txt');

ListBox1.Items.Assign(Lines1);

end;

finally

Lines1.Free;

end;

end;

procedure TForm5.Image6Click(Sender: TObject);

begin

Form5.Hide;

Form1.Show;

Form1.Edit1.Hide;

Form1.ToggleControls(true);

end;

end.