**Введение**

На курсовое проектирование была поставлена задача, разработать программу на тему: Разработка игрового приложения Brick Breaker

Цель курсового проекта заключается в обобщении знаний и подготовки к квалификационному экзамену.

Создаваемое приложение будет рассчитано для развлекательных целей.

Приведем краткое описание разделов пояснительной записки.

Первый раздел носит название “Анализ задачи”. В нем вы сможете ознакомиться с постановкой задачи, которая включает в себя: исследование предметной области поставленной задачи, определение ее организационно-экономической сущности. Также в этом разделе вы сможете узнать о том, как данная задача решается в настоящее время. Все входные и выходные данные тоже будут описаны в первом разделе. В подразделе “Инструменты разработки” будет рассмотрена среда, в которой создается данный курсовой проект. Здесь также будут установлены минимальные и оптимальные требования к аппаратным характеристикам, обеспечивающим правильное функционирование поставленной задачи.

В разделе “Проектирование задачи” будут рассмотрены основные аспекты разработки программного продукта. Здесь можно будет узнать об организации данных в контексте среды разработки. В данном разделе будет четко описан пользовательский интерфейс, составлены алгоритмы процесса обработки информации, описана разработка системы справочной информации.

“Реализация задачи” – это третий раздел пояснительной записки, в котором описываются все элементы и объекты, которые будут использованы при реализации данного приложения. В этом разделе будут четко описаны функции пользователя и их структура. Здесь можно будет найти таблицу, в которой будет представлена полная аннотация файлов используемых в данном проекте.

Четвертый раздел – “Тестирование”. В нем будет описано полное и функциональное тестирование данной программы, т.е. будет оттестирован каждый пункт меню, каждая операция, которая выполняется приложением. Будут смоделированы все возможные действия пользователя при работе с программой, начиная от запуска до выхода.

В разделе “Применение” будет описано назначение, область применения, среда функционирования курсовой программы. Также в нем будет описано использование справочной системы.

“Заключение” будет содержать краткую формулировку задачи, результаты проделанной работы, описание использованных методов и средств, описание степени автоматизации процессов на различных этапах разработки.

В “Списке используемых источников” будет приведен список используемой литературы, нормативно- техническую и другую документацию.

В приложениях к пояснительной записке будет приведен листинг программы с необходимыми комментариями.

Схема работы системы будет представлена в графической части.

**1. Анализ задачи**

**1.1 Постановка задачи**

Темой данного курсового проекта является «Разработка игрового приложения «Brick Breaker»

Игровые приложения созданы для того, чтобы отвлечь человека от сторонних проблем, дать ему хорошее настроение, развить реакцию и терпение. В игры играют люди разных возрастов, от детей до пожилых людей.

Аналогов для такого приложения можно привести очень много.  
 Арканоид – захватывающая аркадная игра, которая уже десятилетиями

радует игроков своей простотой, увлекательностью и непревзойденным драйвом.

Она является одной из самых известных и популярных аркадных игр в истории. В основе игры лежит простая, но захватывающая механика. Вы контролируете платформу, которая расположена внизу игрового поля, а вашей задачей является отбивать мячик, направляя его к различным блокам, расположенным в верхней части экрана. При столкновении с блоками они разрушаются, и ваша цель – полностью очистить игровое поле от них.

Арканоид – игра, которая никогда не перестает радовать и увлекать игроков, независимо от их возраста и игрового опыта.

Периодичность использования данного программного продукта неограниченна. Пользователь может в любое время установить, а также удалить приложение после того как ему надоест игра.

Цель данного курсового проекта – разработать программный продукт, который будет повышать настроение человеку, а также развивать его реакцию и терпение.

За основу программы возьмем из соответствующих источников (сайтов, игр) наиболее полную, достоверную и точную информацию о игре. При реализации проекта необходимо его сделать простым и максимально доступным для пользователей.

Приложение должно располагать своим интерфейсом и функционалом к пользователю, чтобы ему было комфортно.

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

Разрабатываемый программный продукт должен позволять выполнять следующие действия:

* обеспечить быструю и простую навигацию;
* краткую информацию об игре и авторе;
* просмотр справки;
* удовлетворять развлекательные потребности пользователя;

В разрабатываемой программе будут использоваться два вида данных: выходная и постоянная информация.

К выходной информации можно отнести отображение кнопки «Restart»

Постоянной информацией в проекте будут являться кнопки, картинки и др.

**1.2 Инструменты разработки**

Для разработки данного проекта будет выбрана среда Delphi 11, так как

это самое удобная и доступная среда разработки на данный момент. Delphi 11-язык –программирования, относящийся к классу RAD- (Rapid Application Development – «Средство быстрой разработки приложений») средств CASE –тeхнологии. Delphi 11 сделал разработку приложений дляWindows быстрым иприятным процессом. Теперь разрабатывать сложные и интересные проекты можно только одним человеком, использующим Delphi 11

Интерфейс Windows обеспечивает полное перенесение CASE-технологии в интегральную систему поддержки работ по созданию прикладной системы на всех фазах жизненного цикла работы и проектирования системы.

Delphi 11 обладает широким набором возможностей, начиная от проектировщика форм и заканчивая поддержкой всех форматов популярных баз данных. Среда устраняет необходимость программировать такие компоненты Windows общего назначения, как метки, программы и даже диалоговые панели. Работая в Windows, можно видеть одинаковые «объекты» во многих разнообразных приложениях. Диалоговые панели (например, Choose File и Save File) являются примерами многократно-используемых компонентов, встроенных непосредственно в Delphi 11, который позволяет приспособить эти компоненты к имеющийся задаче, чтобы они работали именно так, как требуется создаваемому приложению. Также здесь имеются предварительно-определенные визуальные и не визуальные объекты.

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 11 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. Delphi 11 позволяет создать различные виды программ: консольные приложения, оконные приложения, приложения для работы с Интернетом и базами данных. То есть, Delphi 11 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.

Выгоды от проектирования в среде Windows с помощью Delphi 11:

* устраняется необходимость в повторном вводе данных;
* обеспечивается согласованность проекта и его реализации;
* увеличивается производительность разработки и переносимость программ.

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

Easy Paint tool SAI - это популярная среди художников программа для рисования (в частности для проработки интерфейса и игры «одевалка»). Удобна своей простотой и в то же время большим функционалом. Программа очень сильно распространена в использовании во всем мире и предпочитается рисующими людьми как один из самых удобных инструментов для создания рисунков. В нем можно сохранить файлы картинок во множестве вариантов (.png, .jpg, .bmp и так далее).

Microsoft Word 2019 – редактор текста для написания документации.

Разработка ведется на ноутбуке Samsung. У данного ноутбука следующие

параметры:

* процессор Intel(R) HD Graphics 4000;
* объем ОЗУ 8 гб;
* объем места на SSD – 254 гб ;
* видеоподсистема 1266х768 точек с глубиной цвета 8 Bit;
* ОС – Windows 10 домашняя.

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.

По этой причине особенно актуальной становится разработка программного электронного продукта, который способствовал бы и давал возможность вспомнить и повторить информацию из теоретического курса. Чтобы эта информация лучше усваивалась необходимо подобрать правильный интерфейс.

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. Все окна приложения по возможности должны помещаться на экран полностью, так как использование в процессе работы полос прокруток достаточно неудобно.

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 сбое 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 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.

**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. Входной информацией в разрабатываемой программе будут являться данные. Подходящей структурой данных для хранения элементов является: динамический одномерный массив.

Третьим видом данных является результат программы –отсортированные пользовательские данные. 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 должно заниматься разрабатываемое приложение. Главной задачей будет являться: удовлетворить развлекательные потребности пользователя.

Для реализации задач будут использоваться процедуры. С помощью процедуры будет осуществляться переход на другие формы проекта.

Появление блоков будет разным, что позволяет пользователю разнообразить свой игровой процесс. Также по некоторым блоком требуется попасть несколько раз, что не даёт большой простоты в игре.

В проекте присутствует справка, чтобы пользователь мог узнать о проекте.

**2.3 Описание внешнего пользовательского интерфейса**

Важным при выполнении курсового проекта является простота интерфейса. 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 на рисунке 1 представлена система меню и организация навигации между окнами программы:

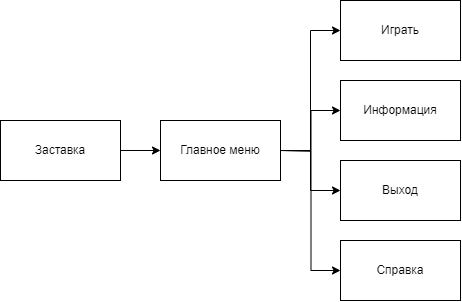


Рисунок 1 – Схема навигации между окнами программы

**3. Реализация**

**3.1 Реализация проекта**

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

Unit 4- является заставкой в курсовом проекте;

Unit 2- является главным меню;

Unit 1- является игрой;

Unit 3- содержит краткую информацию о проекте и авторе;

**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 |
| 1 procedure TForm1.FormCreate(Sender: TObject); | Unit 1 | Form1 | | Создает игровую панель, блоки и устанавливает начальные значения переменных. |
| 2 procedure TForm1.SaveBlockColors | Unit 1 | Form1 | | Сохраняет цвета блоков перед изменением. |
| 3 procedure TForm1.RestoreBlockColors; | Unit 1 | Form1 | | Восстанавливает сохраненные цвета блоков. |
| 4 procedure TForm1.Timer1Timer(Sender: TObject); | Unit 1 | Form1 | | Основная логика игры: перемещение мяча, обработка столкновений с блоками, проверка условий победы или поражения. |
| 5 procedure TForm1.RestartImageClick(Sender: TObject); | Unit 1 | RestartImage | Перезапускает игру. | |
| 6 procedure TForm1.Panel1MouseMove(Sender: TObject; Shift: TShiftState; X, Y: Integer); | Unit 1 | Panel1 | Изменяет положение панели игрока в зависимости от перемещения мыши. | |
| 7 procedure TForm1.Image1Click(Sender: TObject); | Unit 1 | Image1 | Запускает игру | |
| 8 procedure TForm1.Image2Click(Sender: TObject); | Unit 1 | Image2 | Переход на форму 2 | |

|  |  |  |  |
| --- | --- | --- | --- |
| Продолжение таблицы 1 | | | |
| 1 | 2 | 3 | 4 |
| 9 procedure TForm2.FormCreate(Sender: TObject); | Unit 2 | TForm2 | Загружает фоновое изображение, устанавливает его свойства и располагает другие компоненты на форме. |
| 10 procedure TForm2.LoadBackgroundImage(FileName: string); | Unit 2 | TForm2 | Загружает фоновое изображение |
| 11 procedure TForm2.SetBackgroundImageProperties; | Unit 2 | TForm2 | Устанавливает свойства фонового изображения. |
| 12 procedure TForm2.SetBackgroundImageProperties; | Unit 2 | Image2 | Переход на форму 1 |
| 13 procedure TForm2.Image3Click(Sender: TObject); | Unit 2 | Image3 | Переход на форму 3 |
| 14 procedure TForm2.Image6Click(Sender: TObject); | Unit 2 | Image6 | Закрывает проект |
| 15 procedure TForm2.N2Click(Sender: TObject); | Unit 2 | Main | Открывает справку |
| 16 procedure TForm3.FormCreate(Sender: TObject); | Unit 3 | TForm3 | Устанавливает свойства формы, загружает фоновое изображение и устанавливает его свойства, а также устанавливает окно в максимизированное состояние. |
| 17 procedure TForm3.FormResize(Sender: TObject); | Unit 3 | TForm3 | Обработчик события изменения размера формы. Перерасполагает компоненты на форме в зависимости от нового размера окна. |
| 18 procedure TForm3.Image1Click(Sender: TObject); | Unit 3 | Image1 | Переход на форму 2 |
| 19 procedure TForm3.LoadBackgroundImage(FileName: string); | Unit 3 | TForm3 | Загружает фоновое изображение |
| 20 procedure TForm3.SetBackgroundImageProperties; | Unit 3 | TForm3 | Устанавливает свойства фонового изображения |
| 21 procedure TForm4.FormCreate(Sender: TObject); | Unit 4 | TForm4 | Устанавливает стиль формы, включает таймер |
| 22 procedure TForm4.TitleTimerTimer(Sender: TObject); | Unit 4 | TitleTimer | Изменяет прозрачность формы, показывает форму 2 |

**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 |
| TMainMenu | Form2 | Используется для создания главного меню проекта |
| TImage | Form1, Form2, Form3, Form4 | Используется как фон, кнопка |
| TTimer | Form1, Form4 | Используется для отсчета времени |
| TLabel | Form1 | Отображение надписей о поражении и победе |
| TShape | Form1 | Используется как мяч |
| TPanel | Form1 | Используется как рамка для игры, и как платформа |
| TMenuItem | Form2 | Используется для справки |
| TMemo | Form3 | Используется как окно информации |

**3.2 Спецификация программы**

Точное название проекта и его состав приводится в таблице 3.

Таблица 3 – Спецификация программы

|  |  |
| --- | --- |
| Имя файла | Назначение |
| 1 | 2 |
| BrickBreaker.exe | Исполняемый файл проекта, используется для запуска программы на выполнение. |
| BrickBreaker.dproj | Файл проекта, связывает все файлы из которых состоит приложение. |
| Unit4.pas | Файл программного модуля главной формы |
| Unit2.pas | Файл программного модуля главного меню |
| Unit1.pas | Файл программного модуля с игрой |
| Unit3.pas | Файл программного модуля с информацией |
| Unit4.dfm | Главная форма |
| Unit2.dfm | Форма с главным меню |
| Unit1.dfm | Форма с игрой |
| Unit3.dfm | Форма с информацией |
| BrickBreaker.exe | | Установочный пакет программы | |
| bb.jpg | | Изображение | |
| exit.png | | Изображение | |
| info.png | | Изображение | |
| menu.png | | Изображение | |
| play.png | | Изображение | |
| restart.png | | Изображение | |
| Заставка.png | | Изображение | |
| gamehelp.chm | | Справка | |

**4. Тестирование**

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

Отчет о результатах тестирования представлен в таблице 4

Таблица 4- Отчет о результатах тестирования

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Идентификатор теста | Тест | Ожидаемый результат | Физический результат | Результат тестирование |
| 1 | Проверка входа  пользователя в приложение | Открытие главного окна | Открытие главного  окна | Выполнено |
| 2 | Проверка кнопки  «MENU» | Открытие формы с  главным меню | Открытие формы с  главным меню | Выполнено |
| 3 | Проверка кнопки  «PLAY» | Открытие формы с  игрой | Открытие формы с  игрой | Выполнено |
| 4 | Проверка кнопки  «INFO» | Открытие формы с  информацией | Открытие формы с  информацией | Выполнено |
| 5 | Проверка кнопки  «EXIT» | Выход из приложения | Выход из приложения | Выполнено |
| 6 | Проверка кнопки  «RESTART» | Рестарт игры | Рестарт игры | Выполнено |
| 7 | Проверка справки | Открытие справки | Рестарт игры | Выполнено |
| 8 | Проверка движения платформы | Движение платформы | Движение платформы | Выполнено |
| 9 | Проверка движения мяча | Движение мяча | Движение мяча | Выполнено |
| 10 | Проверка исчезновения блоков | Блоки исчезают | Блоки исчезают | Выполнено |

При разработке программного продукта было решено множество проблем, например, не осуществлялся переход к следующей форме по нажатию на изображение или мяч не двигался, поэтому в будущем пользователь не столкнется с данными проблемами.

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

**5 Применение**

**5.1 Общие сведения о программном продукте**

Цель данного проекта заключается в развлекательных целях, улучшения реакции.

Создаваемое приложение будет рассчитано на людей разных вохрастов.

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 Intel(R) Core(TM) i3-3120M CPU @ 2.50GHz, 2500 МГц, ядер: 2, логических процессоров: 4
* объем ОЗУ 8Гб;
* графический адаптер AMD Radeon HD 8600/8700M;
* операционная система Windows 10.

**5.2 Инсталляция**

Для того, чтобы установить программу необходимо запустить файл BrickBreaker.exe. Появится окно установки приложения “ Электронное средство обучения для учащихся профессиональной подготовки рабочих по профессии «Оператор ЭВМ ”.

Затем достаточно следовать приведенной инструкции установки приложения.

**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 «BrickBreaker»

Вторым способом является запуск из каталога, в который устанавливалось приложение (C:\Users\User\Documents\BrickBreaker.exe)

Отчет о результатах тестирования будет представлен в 4 разделе пояснительной записки.

**5.3.2 Инструкции по работе с программой**

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



Рисунок 1 - Заставка

После заставки будет открыто главное меню, представленное на рисунке 2, в котором будет 3 кнопки.

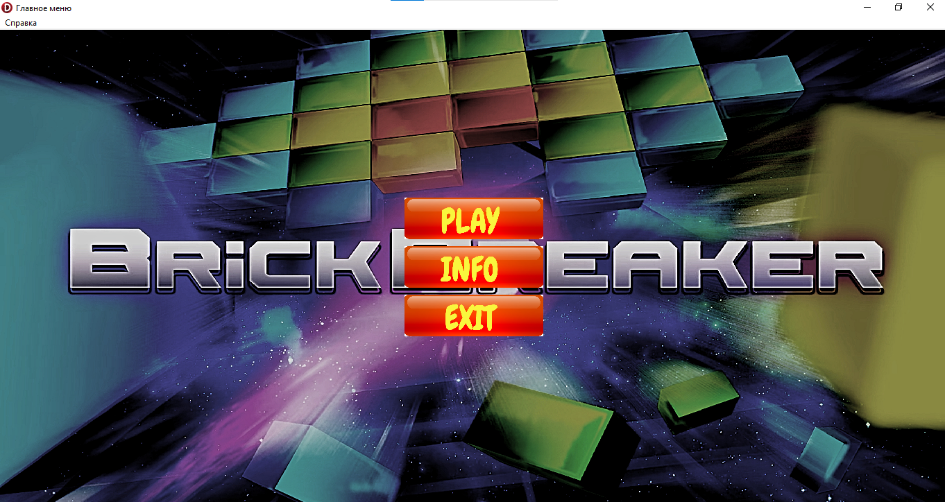


Рисунок 2 - Главное меню

Нажав на кнопку «Play» в главном меню, откроется форма с игрой, представленная на рисунке 3.



Рисунок 3 – Игра

Нажав на кнопку «INFO» в главном меню, откроется форма с информацией, представленная на рисунке 4.

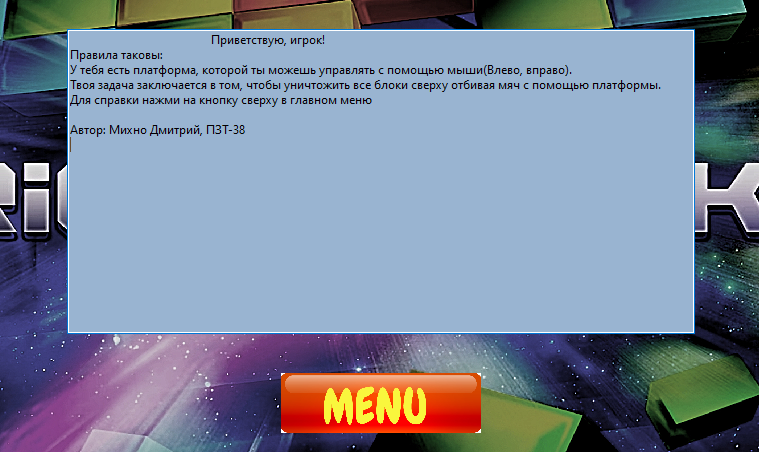
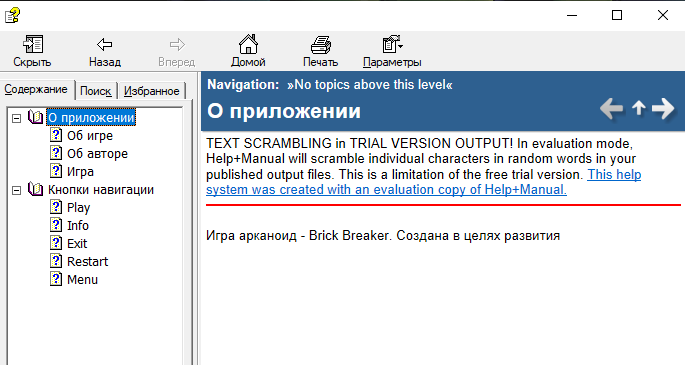


Рисунок 4 - Информация

# 5.4 Использование системы справочной информации

По нажатию на кнопку “Справка” в главном меню откроется справка, представленная на рисунке 5.



# 

Рисунок 5 – Справка

**Заключение**

Целью данного проекта заключалось в развитии реакции, сосредоточенности, терпения и поднятия настроения.

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.

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

**Список использованных источников**

1. Симонович С.В., Евсеев Г.А., Алексеев А.Г. Специальная информатика: Учебное пособие. - М.: АСТ-ПРЕСС: Инфорком-Пресс, 2001. - 950 с.
2. Сайт [www.embarcadero.com](http://www.embarcadero.com) – 3 часа.
3. Вощинская, Г.Э. Разработка компонентов в DELPHI. - Воронеж: ИПЦ ВГУ, 2007. - 57 с.
4. Сайт dzen.ru - Программирование в Delphi 11. – 46 мин.

5. Культин, Н. Delphi 6. Программирование на Object Pascal / Н. Культин. - М.: БХВ-Петербург, 2012. - 1 час.

6. Форум для программистов StackOverflow.com [Электронный ресурс]. – Режим доступа: https://stackoverflow.com. – 2 часа

**Приложение А**

Листинг программы

unit Unit1;

interface

uses

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

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.ExtCtrls, Vcl.StdCtrls, ShellAPI,

Vcl.Imaging.pngimage, Vcl.Imaging.jpeg;

type

TForm1 = class(TForm)

Panel1: TPanel;

Ball: TShape;

Timer1: TTimer;

Image1: TImage;

Image2: TImage;

RestartImage: TImage;

Label4: TLabel;

procedure FormCreate(Sender: TObject);

procedure Timer1Timer(Sender: TObject);

procedure RestartImageClick(Sender: TObject);

procedure Panel1MouseMove(Sender: TObject; Shift: TShiftState; X, Y: Integer);

procedure Image1Click(Sender: TObject);

procedure Image2Click(Sender: TObject);

private

BallXSpeed: Integer;

BallYSpeed: Integer;

FLeftKeyPressed: Boolean;

FRightKeyPressed: Boolean;

FIsButtonPressed: Boolean;

GamePanel: TPanel;

Blocks: array of array of TShape;

BlocksColors: array of array of TColor;

BlockColors: array of array of TColor;

BallFalling: Boolean;

BlocksDestroyed: Integer;

Label2Bottom: Integer;

Label2Visible: Boolean;

TotalBlocks: Integer;

procedure SaveBlockColors;

procedure RestoreBlockColors;

public

{ Public declarations }

end;

var

Form1: TForm1;

implementation

{$R \*.dfm}

uses Unit2;

procedure TForm1.FormCreate(Sender: TObject);

var

HalfScreenWidth: Integer;

I, J: Integer;

BlockColumns, BlockRows: Integer;

BlockWidth, BlockHeight: Integer;

BlockSpacingX, BlockSpacingY: Integer;

begin

Label2Visible := False;

BallFalling := False;

HalfScreenWidth := GetSystemMetrics(SM\_CXSCREEN) div 2;

BlockColumns := 10;

BlockRows := 6;

BlockWidth := (HalfScreenWidth - 40 - (8 \* (BlockColumns + 1))) div BlockColumns;

BlockHeight := 20;

BlockSpacingX := 8;

BlockSpacingY := 8;

GamePanel := TPanel.Create(Self);

GamePanel.Parent := Form1;

GamePanel.Width := HalfScreenWidth;

GamePanel.Height := ClientHeight;

GamePanel.Left := 0;

GamePanel.BorderStyle := bsSingle;

GamePanel.BevelOuter := bvLowered;

GamePanel.Color := clBlack;

SetLength(Blocks, BlockColumns, BlockRows);

SetLength(BlocksColors, BlockColumns, BlockRows);

SetLength(BlockColors, BlockColumns, BlockRows);

BlocksDestroyed := 0;

TotalBlocks := BlockColumns \* BlockRows;

for I := 0 to BlockColumns - 1 do

begin

for J := 0 to BlockRows - 1 do

begin

Blocks[I, J] := TShape.Create(Self);

Blocks[I, J].Parent := GamePanel;

Blocks[I, J].Width := BlockWidth;

Blocks[I, J].Height := BlockHeight;

Blocks[I, J].Shape := stRectangle;

Blocks[I, J].Left := BlockSpacingX + (BlockWidth + BlockSpacingX) \* I;

Blocks[I, J].Top := BlockSpacingY + (BlockHeight + BlockSpacingY) \* J;

case Random(3) of

0:

begin

BlocksColors[I, J] := clRed;

BlockColors[I, J] := clRed;

end;

1:

begin

BlocksColors[I, J] := clGreen;

BlockColors[I, J] := clGreen;

end;

2:

begin

BlocksColors[I, J] := clBlack;

BlockColors[I, J] := clBlack;

end;

end;

Blocks[I, J].Brush.Color := BlocksColors[I, J];

end;

end;

Ball.Parent := GamePanel;

Panel1.Parent := GamePanel;

GamePanel.Left := (HalfScreenWidth - GamePanel.Width) div 2;

Image1.Parent := Form1;

Image1.Left := GamePanel.Left + GamePanel.Width + 10;

Image1.Top := (ClientHeight - (Image1.Height + Image2.Height + RestartImage.Height + 20)) div 2;

Image2.Parent := Form1;

Image2.Left := Image1.Left;

Image2.Top := Image1.Top + Image1.Height + 10;

RestartImage.Parent := Form1;

RestartImage.Left := Image1.Left;

RestartImage.Top := Image2.Top + Image2.Height + 10;

Panel1.Left := (GamePanel.Width - Panel1.Width) div 2;

Panel1.Top := GamePanel.Height - Panel1.Height - 10;

BallXSpeed := 5;

BallYSpeed := -5;

FLeftKeyPressed := False;

FRightKeyPressed := False;

Timer1.Enabled := True;

RestartImage.Visible := False;

end;

procedure TForm1.SaveBlockColors;

var

I, J: Integer;

begin

for I := 0 to Length(BlocksColors) - 1 do

begin

for J := 0 to Length(BlocksColors[I]) - 1 do

begin

BlockColors[I, J] := BlocksColors[I, J];

end;

end;

end;

procedure TForm1.RestoreBlockColors;

var

I, J: Integer;

begin

for I := 0 to Length(BlocksColors) - 1 do

begin

for J := 0 to Length(BlocksColors[I]) - 1 do

begin

BlocksColors[I, J] := BlockColors[I, J];

Blocks[I, J].Brush.Color := BlocksColors[I, J];

end;

end;

end;

procedure TForm1.Timer1Timer(Sender: TObject);

var

MousePos: Integer;

I, J: Integer;

begin

MousePos := Mouse.CursorPos.X - GamePanel.Left - (Panel1.Width div 2);

if MousePos < 0 then

MousePos := 0

else if MousePos > GamePanel.Width - Panel1.Width then

MousePos := GamePanel.Width - Panel1.Width;

Panel1.Left := MousePos;

if FIsButtonPressed then

begin

if Ball.Left <= GamePanel.Left then

BallXSpeed := Abs(BallXSpeed)

else if Ball.Left + Ball.Width >= GamePanel.Left + GamePanel.Width then

BallXSpeed := -Abs(BallXSpeed);

if Ball.Top <= GamePanel.Top then

BallYSpeed := Abs(BallYSpeed)

else if Ball.BoundsRect.IntersectsWith(Panel1.BoundsRect) then

begin

if not BallFalling then

begin

BallYSpeed := 5;

BallFalling := True;

end;

if Ball.Top + Ball.Height >= Panel1.Top then

begin

Ball.Top := Panel1.Top - Ball.Height;

BallYSpeed := -Abs(BallYSpeed);

end;

end

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

begin

Label4.Parent := GamePanel;

Label4.Caption := 'Вы проиграли';

Label4.Alignment := taCenter;

Label4.Left := (GamePanel.Width - Label4.Width) div 2;

Label4.Top := (GamePanel.Height - Label4.Height) div 2;

Label4.Visible := True;

SaveBlockColors;

RestoreBlockColors;

RestartImage.Visible := True;

Image2.Visible := True;

Timer1.Enabled := False;

BallFalling := False;

end;

Ball.Left := Ball.Left + BallXSpeed;

Ball.Top := Ball.Top + BallYSpeed;

for I := 0 to Length(Blocks) - 1 do

begin

for J := 0 to Length(Blocks[I]) - 1 do

begin

if Blocks[I, J].Visible and Ball.BoundsRect.IntersectsWith(Blocks[I, J].BoundsRect) then

begin

if Blocks[I, J].Brush.Color = clRed then

begin

Blocks[I, J].Visible := False;

BallYSpeed := -BallYSpeed;

Inc(BlocksDestroyed);

end

else if Blocks[I, J].Brush.Color = clGreen then

begin

Blocks[I, J].Brush.Color := clRed;

BallYSpeed := -BallYSpeed;

end

else if Blocks[I, J].Brush.Color = clBlack then

begin

Blocks[I, J].Brush.Color := clGreen;

BallYSpeed := -BallYSpeed;

end;

end;

end;

end;

if BlocksDestroyed = TotalBlocks then

begin

Label4.Parent := GamePanel;

Label4.Caption := 'Поздравляем, вы выиграли!';

Label4.Alignment := taCenter;

Label4.Left := (GamePanel.Width - Label4.Width) div 2;

Label4.Top := (GamePanel.Height - Label4.Height) div 2;

Label4.Visible := True;

SaveBlockColors;

RestoreBlockColors;

RestartImage.Visible := True;

Image2.Visible := True;

Timer1.Enabled := False;

BallFalling := False;

end;

end;

end;

procedure TForm1.RestartImageClick(Sender: TObject);

var

I, J: Integer;

begin

Label4.Visible := False;

RestartImage.Visible := False;

Image2.Visible := False;

for I := 0 to Length(Blocks) - 1 do

begin

for J := 0 to Length(Blocks[I]) - 1 do

begin

if Blocks[I, J] <> nil then

begin

Blocks[I, J].Visible := True;

end;

end;

end;

Ball.Left := Panel1.Left + (Panel1.Width div 2) - (Ball.Width div 2);

Ball.Top := Panel1.Top - Ball.Height - 1;

BallYSpeed := -5;

BallFalling := False;

BlocksDestroyed := 0;

Timer1.Enabled := True;

end;

procedure TForm1.Panel1MouseMove(Sender: TObject; Shift: TShiftState; X,

Y: Integer);

begin

if not FIsButtonPressed then

begin

if (X > Panel1.Width div 2) and not FRightKeyPressed then

begin

FRightKeyPressed := True;

FLeftKeyPressed := False;

end

else if (X < Panel1.Width div 2) and not FLeftKeyPressed then

begin

FLeftKeyPressed := True;

FRightKeyPressed := False;

end;

if FLeftKeyPressed then

begin

Panel1.Left := Panel1.Left - 10;

if Panel1.Left < 0 then

Panel1.Left := 0;

end

else if FRightKeyPressed then

begin

Panel1.Left := Panel1.Left + 10;

if Panel1.Left > GamePanel.Width - Panel1.Width then

Panel1.Left := GamePanel.Width - Panel1.Width;

end;

end;

end;

procedure TForm1.Image1Click(Sender: TObject);

begin

BallXSpeed := 5;

BallYSpeed := -5;

FIsButtonPressed := True;

Image2.Visible := False;

Image1.Visible := False;

end;

procedure TForm1.Image2Click(Sender: TObject);

begin

Form2.Show;

Hide;

end;

end.

unit Unit2;

interface

uses

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

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

Vcl.StdCtrls, Vcl.Imaging.pngimage, ShellAPI, Vcl.Menus;

type

TForm2 = class(TForm)

Image1: TImage;

Image2: TImage;

Image3: TImage;

Image6: TImage;

MainMenu1: TMainMenu;

N2: TMenuItem;

procedure FormCreate(Sender: TObject);

procedure Image2Click(Sender: TObject);

procedure Image3Click(Sender: TObject);

procedure Image6Click(Sender: TObject);

procedure N2Click(Sender: TObject);

private

procedure LoadBackgroundImage(FileName: string);

procedure SetBackgroundImageProperties;

public

{ Public declarations }

end;

var

Form2: TForm2;

implementation

{$R \*.dfm}

uses Unit1, Unit3, Unit4;

procedure TForm2.LoadBackgroundImage(FileName: string);

begin

Image1.Picture.LoadFromFile(FileName);

end;

procedure TForm2.N2Click(Sender: TObject);

begin

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

end;

procedure TForm2.SetBackgroundImageProperties;

begin

Image1.Align := alClient;

Image1.Stretch := True;

end;

procedure TForm2.FormCreate(Sender: TObject);

var

ImageSpacing: Integer;

begin

LoadBackgroundImage('bb.jpg');

SetBackgroundImageProperties;

Form2.Position := poScreenCenter;

ImageSpacing := 10;

Image2.Top := (ClientHeight - Image2.Height - Image3.Height - Image6.Height - (2 \* ImageSpacing)) div 2;

Image2.Left := (ClientWidth - Image2.Width) div 2;

Image3.Top := Image2.Top + Image2.Height + ImageSpacing;

Image3.Left := (ClientWidth - Image3.Width) div 2;

Image6.Top := Image3.Top + Image3.Height + ImageSpacing;

Image6.Left := (ClientWidth - Image6.Width) div 2;

end;

procedure TForm2.Image2Click(Sender: TObject);

begin

Form1.Show;

Hide;

end;

procedure TForm2.Image3Click(Sender: TObject);

begin

Form3.Show;

Hide;

end;

procedure TForm2.Image6Click(Sender: TObject);

begin

Close;

end;

end.

unit Unit3;

interface

uses

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

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

Vcl.Imaging.jpeg, Vcl.ExtCtrls, ShellAPI;

type

TForm3 = class(TForm)

Memo1: TMemo;

Image2: TImage;

Image1: TImage;

procedure Image1Click(Sender: TObject);

procedure FormResize(Sender: TObject);

procedure FormCreate(Sender: TObject);

private

{ Private declarations }

procedure LoadBackgroundImage(FileName: string);

procedure SetBackgroundImageProperties;

public

{ Public declarations }

end;

var

Form3: TForm3;

implementation

{$R \*.dfm}

uses

Unit1, Unit2;

procedure TForm3.FormCreate(Sender: TObject);

begin

Form3.BorderStyle := bsSizeable;

Form3.Position := poScreenCenter;

LoadBackgroundImage('bb.jpg');

SetBackgroundImageProperties;

Image1.Transparent := True;

Form3.WindowState := wsMaximized;

end;

procedure TForm3.FormResize(Sender: TObject);

begin

Memo1.Left := (ClientWidth - Memo1.Width) div 2;

Memo1.Top := (ClientHeight div 2) - (Memo1.Height div 2) - 20;

Image1.Left := (ClientWidth - Image1.Width) div 2;

Image1.Top := (ClientHeight div 2) + (Memo1.Height div 2) + 20;

end;

procedure TForm3.Image1Click(Sender: TObject);

begin

Form2.Show;

Hide;

end;

procedure TForm3.LoadBackgroundImage(FileName: string);

begin

Image2.Picture.LoadFromFile(FileName);

end;

procedure TForm3.SetBackgroundImageProperties;

begin

Image2.Align := alClient;

Image2.Stretch := True;

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.Imaging.pngimage, Vcl.ExtCtrls;

type

TForm4 = class(TForm)

Image1: TImage;

TitleTimer: TTimer;

procedure FormCreate(Sender: TObject);

procedure TitleTimerTimer(Sender: TObject);

private

{ Private declarations }

x: Integer;

public

{ Public declarations }

end;

var

Form4: TForm4;

implementation

{$R \*.dfm}

Uses

Unit2;

procedure TForm4.FormCreate(Sender: TObject);

begin

FormStyle := fsStayOnTop;

TitleTimer.Enabled := True;

x := 0;

end;

procedure TForm4.TitleTimerTimer(Sender: TObject);

begin

if AlphaBlendValue = 255 then

x := x + 1;

if x = 0 then

AlphaBlendValue := AlphaBlendValue + 5

else if x = 100 then

AlphaBlendValue := AlphaBlendValue - 5;

if AlphaBlendValue = 0 then

begin

Form2.Show;

TitleTimer.Enabled := False;

Hide;

end;

end;

end.