**Введение**

Задача данного проекта является создание приложения, которое будет рассказывать о городских скульптурах Гродно. В рамках проекта планируется привлечение внимания к культурному наследию города.

Повышение осведомленности о его исторических и художественных достоинствах является его основной целью.

Разработка данного проекта сделает эту информацию более доступной и понятной для широкой аудитории, способствовать повышению культурной осведомленности, укреплять городскую идентичность.

Описание разделов пояснительной задачи:

Первый раздел, озаглавленный "Анализ задачи", включает исследование предметной области и определение организационно-экономической сущности задачи. В этом разделе также описаны текущие методы решения задачи, а также входные и выходные данные. Подраздел "Инструменты разработки" посвящен описанию среды разработки курсового проекта и минимальным и оптимальным требованиям к аппаратным характеристикам для его корректной работы.

В разделе "Проектирование задачи" рассматриваются ключевые аспекты разработки программного продукта. Здесь описывается организация данных в среде разработки, пользовательский интерфейс, алгоритмы обработки информации и создание справочной системы.

Третий раздел, "Реализация задачи", описывает все элементы и объекты, используемые при реализации приложения. В этом разделе также подробно рассмотрены функции пользователя и их структура, а также представлена таблица с аннотацией всех файлов проекта.

Четвертый раздел, "Тестирование", посвящен полному и функциональному тестированию программы. В нем проверяется каждый пункт меню и операция приложения, моделируются все возможные действия пользователя от запуска до выхода.

Раздел "Применение" описывает назначение, область применения и среду функционирования курсового проекта, а также использование справочной системы.

В разделе "Заключение" подводятся итоги задачи, результаты работы, описание использованных методов и средств, а также степень автоматизации процессов на различных этапах разработки.

Раздел "Литература" содержит перечень источников, использованных при разработке проекта.

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**

В рамках данной предметной области рассматриваются различные аспекты исторических, культурных и художественных объектов, находящихся в городской среде Гродно. Эти скульптуры являются неотъемлемой частью культурного наследия города и привлекают туристов и жителей города. Городские скульптуры Гродно включают множество памятников и художественных объектов, представляющих историческое и культурное значение. Они отражают различные эпохи и стили, являясь свидетелями богатой истории города. Каждая скульптура имеет свою уникальную историю создания, которая рассказывает о жизни города и его обитателей. Эти объекты не только украшают город, но и играют важную роль в сохранении и популяризации культурного наследия Гродно. Скульптуры, памятники и другие художественные объекты, расположенные в различных частях города, служат не только украшением городской среды, но и важными элементами культурного кода города. Они создают уникальную атмосферу, привлекая внимание жителей и гостей города к истории и культуре Гродно. Основной целевой аудиторией являются туристы и гости города, которые приезжают, чтобы познакомиться с культурой и историей Гродно. Однако, приложение будет также полезно для жителей города, интересующихся историей и культурой своего города, а также для студентов, изучающих искусство, архитектуру и историю. Туристы смогут использовать приложение для посещения достопримечательностей, а жители города - для углубленного изучения истории родного города. Студенты смогут использовать приложение в учебных и научных целях, получая доступ к подробной и структурированной информации.

Основной целевой аудиторией являются туристы и гости города, которые приезжают, чтобы познакомиться с культурой и историей Гродно. Однако, приложение будет также полезно для жителей города, интересующихся историей и культурой своего города, а также для студентов, изучающих искусство, архитектуру и историю. Туристы смогут использовать приложение для посещения достопримечательностей, а жители города - для углубленного изучения истории родного города. Студенты смогут использовать приложение в учебных и научных целях, получая доступ к подробной и структурированной информации. Помимо этого, приложение может быть полезно исследователям, историкам и культурологам, которые изучают культурное наследие города и его историческое развитие. Семьи с детьми также могут использовать приложение для образовательных прогулок и экскурсий по городу.

В настоящее время информация о городских скульптурах Гродно доступна в виде печатных и электронных путеводителей, сайтов и блогов, посвященных истории города. Местные музеи и туристические информационные центры предоставляют частичную информацию. Локальные экскурсии с гидами также позволяют углубиться в историю города. Однако, данные источники не всегда актуальны и не предоставляют полной информации в удобной и доступной форме. Приложение решит эту проблему, обеспечивая актуальную и полную информацию в любое время и в любом месте.

Периодичность использования программы будет зависеть лишь от потребностей пользователей. Туристы будут использовать приложение в ходе своих поездок и при планировании маршрутов, жители города - для изучения истории и культуры города в любое удобное время, а студенты для учебных и научных целей. Приложение будет востребовано как в будние дни, так и в выходные, в зависимости от времени и интересов пользователей. Семьи с детьми могут использовать приложение в выходные дни для образовательных прогулок, студенты - в ходе учебных занятий и подготовки к экзаменам, а туристы - в любое время, когда они находятся в городе. Приложение будет полезно также для экскурсоводов и гидов, которые смогут использовать его для подготовки и проведения экскурсий, а также для представления дополнительной информации своим клиентам.

Новое приложение будет упрощать доступ к информации о городских скульптурах, делая её легко доступной в любом месте и в любое время. Приложение будет предоставлять актуальную информацию, что позволит пользователям исследовать скульптуры с помощью карты. Пользователи смогут легко находить интересующие их объекты, получать полную информацию о них и планировать свои маршруты. Это повысит интерес к культурному наследию города и сделает его более доступным для широкой аудитории.

Аналогами данного приложения могут являться локальные туристические приложения, ведь некоторые города имеют свои собственные приложения, предоставляющие информацию о достопримечательностях.

Выходная информация для данного проекта включает текстовую информацию: описание скульптур, историческую справку, информацию об авторе, дате создания и значении объекта. Изображения: фотографии скульптур, исторические снимки, карты местоположения. Эти данные будут представлены в удобном и доступном формате, что позволит пользователям легко находить интересующую их информацию и использовать её в своих целях. Приложение будет поддерживать различные форматы данных, что сделает информацию более доступной и понятной для пользователей.

К входной информации можно отнести выбранные пользователем значения, например, выбор ответов на вопросы при прохождении теста.

Постоянная информация представляет собой каталог скульптур, описания, изображения и другие данные, которые не изменяются часто. Эта информация будет храниться в базе данных и использоваться для предоставления актуальной и полной информации пользователям. Каталог будет регулярно обновляться, чтобы обеспечить актуальность данных.

Требованиями к выполняемым функциям проекта являются отображение всех скульптур на карте, возможность проходить тест для закрепления изученного материала и предоставление краткой и четкой информации о скульптурах. Карта позволит пользователям легко находить интересующие их объекты и планировать свои маршруты. Тесты помогут пользователям проверить свои знания и улучшить их. Краткая и четкая информация о скульптурах сделает приложение удобным и полезным для всех категорий пользователей.

**1.2 Инструменты разработки**

Для разработки данного проекта будет выбрана среда Delphi 11. Он известен своей мощной функциональностью, визуальным конструктором форм и возможностью быстрого создания приложений для различных платформ.

Delphi предоставляет визуальные инструменты для быстрого создания пользовательского интерфейса (UI), что значительно ускоряет процесс разработки. Большое количество готовых компонентов и библиотек упрощает интеграцию различных функций. Delphi имеет встроенную поддержку работы с различными базами данных, включая MySQL, PostgreSQL, SQLite и другие. Это важно для хранения и управления информацией о скульптурах, событиях и пользователях. 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. Большое и активное сообщество разработчиков Delphi обеспечивает доступ к множеству ресурсов, библиотек и примеров кода. Регулярные обновления и поддержка от компании Embarcadero Technologies гарантируют актуальность и надежность инструментов разработки.

Язык программирования Object Pascal обладает сильной типизацией и объектно-ориентированным подходом. Понятный синтаксис и низкий порог вхождения для разработчиков с опытом в других языках программирования. Позволяет создавать пользовательский интерфейс с помощью перетаскивания компонентов на форму. Поддержка различных стандартных и пользовательских компонентов (кнопки, поля ввода, списки, карты и т.д.). Также возможность создания нативных приложений для различных платформ из единого исходного кода. Поддержка адаптивного интерфейса для различных экранов и устройств. Обширная официальная документация и большое количество учебных материалов, форумов и сообществ разработчиков.

Smart Install Maker — это инструмент для создания инсталляционных пакетов для программного обеспечения. Интуитивно понятный интерфейс позволяет быстро создавать установочные пакеты даже без глубоких знаний в программировании. Сжатие файлов и оптимизация позволяет создавать компактные инсталляционные пакеты, что ускоряет их загрузку и установку и поддержка широкого спектра параметров настройки, включая добавление ярлыков, редактирование реестра, копирование файлов и создание пользовательских диалогов. Возможность включения различных типов файлов в инсталляционные пакеты, таких как библиотеки, динамически подключаемые модули и шрифты.

Easy Paint tool SAI — это популярный графический редактор, разработанный японской компанией SYSTEMAX. Программа широко используется художниками и иллюстраторами благодаря своим многочисленным преимуществам и уникальным функциям. Простота и удобство в использовании делают программу доступной для начинающих и профессионалов. Программа занимает мало места на диске и требует минимальных системных ресурсов, что позволяет работать даже на менее мощных компьютерах без потери производительности. Возможность импорта и экспорта популярных графических форматов, таких как PSD, BMP, JPG и PNG, что обеспечивает совместимость с другими графическими редакторами. Удобные цветовые палитры и инструменты для смешивания цветов позволяют легко подбирать и изменять цвета, добиваясь желаемого результата. Программа отличается высокой надежностью и стабильностью, минимизируя вероятность сбоев и потери данных в процессе работы.

Microsoft Word 2019 – редактор текста для написания документации.

Разработка ведется на ноутбуке Acer. У данного ноутбука следующие

параметры:

* процессор Intel Pentium N3710;
* объем ОЗУ 4 гб;
* объем места на HDD – 256 гб;
* видеоподсистема 1024х768 точек с глубиной цвета 16 Bit;
* ОС – Windows 10 домашняя.

Приложение обладает низкими требованиями к аппаратным ресурсам, что является значительным преимуществом.

**1.3 Требования к приложению**

Приложение "Городские скульптуры Гродно" должно соответствовать ряду технических и функциональных ограничений, которые обеспечат его стабильную работу и удобство использования. Приложение должно быть оптимизировано для работы на устройствах с ограниченными ресурсами. Приложение должно обеспечивать быстрое время отклика.

Интерфейс приложения должен быть интуитивно понятным, удобным и соответствовать современным стандартам. Приложение должно иметь удобную навигацию с использованием стандартных элементов интерфейса, таких как меню, вкладки и кнопки. 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. Карта города с отмеченными на ней скульптурами должна быть легко читаемой.

Специальных требований к надежности не предъявляется. Время восстановления после отказа, вызванного сбоем электропитания технических средств (иными внешними факторами), не фатальным сбое (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.

**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, кнопки), описание, исходные коды теории в разделах.

Вторым видом данных, используемых в программе, является вводимая пользователем информация. К входной информации можно отнести выбираемые пользователем значения, например ответы на вопросы при прохождении теста.

Третьим видом данных является результат программы – предоставление пользователю оценку за тест в виде процентного соотношения.

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 приложение. Главной задачей будет являться: углубленное изучение истории, культуры и художественного наследия города.

Для реализации задач будут использоваться процедуры. С помощью процедуры будет осуществляться переход на текстовый файл в формате .docx.

Тест будет генерироваться на основе выбора пункта в объекте TradioGroup. Вопросы теста будут подгружаться из .txt файла. Далее после того, как будет прочитан теоретический материал, можно будет пройти тест для проверки своих знаний и получить оценку своих знаний.

**2.3 Описание внешнего пользовательского интерфейса**

Важным аспектом при выполнении курсового проекта является обеспечение эффективного взаимодействия между пользователем и программой. Успешная реализация этой задачи во многом зависит от того, как программист спроектирует и разработает приложение, какие компоненты будут использованы и каким образом будут автоматизированы методы взаимодействия. Особое внимание следует уделить пользовательскому интерфейсу. Программисту необходимо организовать внешний вид приложения таким образом, чтобы пользователю было интуитивно понятно, какие действия от него требуются. Это означает, что интерфейс должен быть логичным, последовательным и простым в использовании. Каждый элемент интерфейса должен быть размещен и оформлен таким образом, чтобы пользователю не приходилось долго разбираться, как с ним взаимодействовать. Для обеспечения эффективной работы пользователя необходимо создать целостное приложение, охватывающее все аспекты предметной области. Это означает, что все компоненты приложения должны быть четко сгруппированы по функциональному назначению. Например, разделы для поиска информации о скульптурах, интерактивная карта, образовательные тесты и пользовательские настройки должны быть логически организованы и легко доступны. При этом важно обеспечить удобный графический интерфейс. Пользователь должен иметь возможность решать свои задачи быстро и с минимальными усилиями. Это особенно важно в контексте работы с обширной и разрозненной информацией, такой как данные о городских скульптурах и памятниках. Ниже на рисунке 1 представлена система меню и организация навигации между окнами программы:



Рисунок 1 – Схема навигация между окнами программы

**3. Реализация**

**3.1 Реализация проекта**

Данный курсовой проект содержит 16 модулей. Далее рассмотрим назначение каждого модуля:

Zastavka – является заставкой в курсовом проекте;

MainMenu – является главным меню;

LoadingR – является загрузкой с главного меню на карту Гродно;

LoadingL – является загрузкой с карты Гродно в главное меню;

Unit 5 – карта Гродно;

Unit 6 – содержит информацию о скульптуре «Горожанин 18 века»;

Unit 7 – содержит информацию о скульптуре «Диалог»;

Unit 8 – содержит информацию о скульптуре «Лягушка-путешественница»;

Unit 9 – содержит информацию о скульптуре «Почтовый ящик»;

Unit 10 – содержит информацию о скульптуре «Петрушка»;

Unit 11 – содержит информацию о скульптуре «Верность»;

Unit 12 – содержит информацию о скульптуре «Колонна-маяк»;

Unit 13 – содержит информацию о скульптуре «Ладья-нассад»;

Unit 14 – содержит информацию о скульптуре «Дорожный рабочий»;

Unit 15 – содержит информацию о скульптуре «Погоня на Грюнвальд»;

Unit 16 – тестовые задания;

**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 TzZastavka.FormCreate(Sender: TObject); | Zastavka | zZastavka | | Запуск определенной заставки |
| 2 procedure TzZastavka.Timer1Timer(Sender: TObject); | Zastavka | Timer1 | | Переход на следующую форму |
| 3 procedure TzZastavka.FormDestroy(Sender: TObject); | Zastavka | zZastavka | | Удаление шрифта |
| Продолжение таблицы 1 | | | | |
| 4 procedure TmMainMenu.BitBtn1Click(Sender: TObject); | MainMenu | BitBtn1 | | Переход на следующую форму |
| 5 procedure TmMainMenu.BitBtn3Click(Sender: TObject); | MainMenu | BitBtn3 | Переход на форму с тестом | |
| 6 procedure TmMainMenu.BitBtn4Click(Sender: TObject); | MainMenu | BitBtn4 | Выход | |
| 7 procedure TmMainMenu.N2Click(Sender: TObject); | MainMenu | Main | Выход | |
| 8 procedure TmMainMenu.N3Click(Sender: TObject); | MainMenu | Main | Показ информации об авторе | |
| 9 procedure TmMainMenu.N4Click(Sender: TObject); | MainMenu | Main | Показ справки | |
| 10 procedure TmMainMenu.FormCreate(Sender: TObject); | MainMenu | mMainMenu | Загрузка шрифта | |
| 11 procedure TmMainMenu.FormDestroy(Sender: TObject); | MainMenu | mMainMenu | Удаление шрифта | |
| 12 procedure TlLoadingR.FormCreate(Sender: TObject); | LoadingR | lLoadingR | Загрузка шрифта | |
| 13 procedure TlLoadingR.FormDestroy(Sender: TObject); | LoadingR | lLoadingR | Удаление шрифта | |
| 14 procedure TlLoadingR.FormHide(Sender: TObject); | LoadingR | lLoadingR | Скрыть таймер | |
| 15 procedure TlLoadingR.FormShow(Sender: TObject); | LoadingR | lLoadingR | Запуск определенной загрузки | |
| 16 procedure TlLoadingR.Timer2Timer(Sender: TObject); | LoadingR | lLoadingR | Переход на следующую форму | |
| 17 procedure TlLoadingL.FormCreate(Sender: TObject); | LoadingL | lLoadingL | Загрузка шрифта | |
| 18 procedure TlLoadingL.FormDestroy(Sender: TObject); | LoadingL | lLoadingL | Удаление шрифта | |
| 19 procedure TlLoadingL.FormHide(Sender: TObject); | LoadingL | lLoadingL | Скрыть таймер | |
| 20 procedure TlLoadingL.FormShow(Sender: TObject); | LoadingL | lLoadingL | Запуск определенной загрузки | |
| 21 procedure TlLoadingL.Timer3Timer(Sender: TObject); | LoadingL | Timer3 | Переход на предыдущую форму | |
| 22 procedure TForm5.FormCreate(Sender: TObject); | Unit 5 | Form5 | Загрузка шрифта | |
| 23 procedure TForm5.FormDestroy(Sender: TObject); | Unit 5 | Form5 | Удаление шрифта | |
| 24 procedure TForm5.BitBtn1Click(Sender: TObject); | Unit 5 | BitBtn1 | Переход на предыдущую форму | |
| 25 procedure TForm5.BitBtn2Click(Sender: TObject); | Unit 5 | BitBtn2 | Переход на следующую форму | |
| 26 procedure TForm5.Image1Click(Sender: TObject); | Unit 5 | Image1 | Показ информации о скульптуре | |
| Продолжение таблицы 1 | | | | |
| 27 procedure TForm5.Image2Click(Sender: TObject); | Unit 5 | Image2 | Показ информации о скульптуре | |
| 28 procedure TForm5.Image3Click(Sender: TObject); | Unit 5 | Image3 | Показ информации о скульптуре | |
| 29 procedure TForm5.Image4Click(Sender: TObject); | Unit 5 | Image4 | Показ информации о скульптуре | |
| 30 procedure TForm5.Image5Click(Sender: TObject); | Unit 5 | Image5 | Показ информации о скульптуре | |
| 31 procedure TForm5.Image6Click(Sender: TObject); | Unit 5 | Image6 | Показ информации о скульптуре | |
| 32 procedure TForm5.Image7Click(Sender: TObject); | Unit 5 | Image7 | Показ информации о скульптуре | |
| 33 procedure TForm5.Image8Click(Sender: TObject); | Unit 5 | Image8 | Показ информации о скульптуре | |
| 34 procedure TForm5.Image9Click(Sender: TObject); | Unit 5 | Image9 | Показ информации о скульптуре | |
| 35 procedure TForm5.Image10Click(Sender: TObject); | Unit 5 | Image10 | Показ информации о скульптуре | |
| 36 procedure TForm5.Label1Click(Sender: TObject); | Unit 5 | Label1 | Переход на форму с информацией о скульптуре | |
| 37 procedure TForm5.Label2Click(Sender: TObject); | Unit 5 | Label2 | Переход на форму с информацией о скульптуре | |
| 38 procedure TForm5.Label3Click(Sender: TObject); | Unit 5 | Label3 | Переход на форму с информацией о скульптуре | |
| 39 procedure TForm5.Label4Click(Sender: TObject); | Unit 5 | Label4 | Переход на форму с информацией о скульптуре | |
| 40 procedure TForm5.Label5Click(Sender: TObject); | Unit 5 | Label5 | Переход на форму с информацией о скульптуре | |
| 41 procedure TForm5.Label6Click(Sender: TObject); | Unit 5 | Label6 | Переход на форму с информацией о скульптуре | |
| 42 procedure TForm5.Label7Click(Sender: TObject); | Unit 5 | Label7 | Переход на форму с информацией о скульптуре | |
| 43 procedure TForm5.Label8Click(Sender: TObject); | Unit 5 | Label8 | Переход на форму с информацией о скульптуре | |
| 44 procedure TForm5.Label9Click(Sender: TObject); | Unit 5 | Label9 | Переход на форму с информацией о скульптуре | |
| 45 procedure TForm5.Label10Click(Sender: TObject); | Unit 5 | Label10 | Переход на форму с информацией о скульптуре | |
| Продолжение таблицы 1 | | | | |
| 46 procedure TZhiliber.BitBtn1Click(Sender: TObject); | Unit 6 | BitBtn1 | Переход на предыдущую форму | |
| 47 procedure TZhiliber.FormCreate(Sender: TObject); | Unit 6 | Zhiliber | Загрузка шрифта | |
| 48 procedure TZhiliber.FormDestroy(Sender: TObject); | Unit 6 | Zhiliber | Удаление шрифта | |
| 49 procedure TDialogue.BitBtn1Click(Sender: TObject); | Unit 7 | BitBtn1 | Переход на предыдущую форму | |
| 50 procedure TDialogue.FormCreate(Sender: TObject); | Unit 7 | Dialogue | Загрузка шрифта | |
| 51 procedure TDialogue.FormDestroy(Sender: TObject); | Unit 7 | Dialogue | Удаление шрифта | |
| 52 procedure TFrog.BitBtn1Click(Sender: TObject); | Unit8 | BitBtn1 | Переход на предыдущую форму | |
| 53 procedure TFrog.FormCreate(Sender: TObject); | Unit 8 | Frog | Загрузка шрифта | |
| 54 procedure TFrog.FormDestroy(Sender: TObject); | Unit 8 | Frog | Удаление шрифта | |
| 55 procedure TMail.BitBtn1Click(Sender: TObject); | Unit 9 | BitBtn1 | Переход на предыдущую форму | |
| 56 procedure TMail.FormCreate(Sender: TObject); | Unit 9 | Mail | Загрузка шрифта | |
| 57 procedure TMail.FormDestroy(Sender: TObject); | Unit 9 | Mail | Удаление шрифта | |
| 58 procedure TClown.BitBtn1Click(Sender: TObject); | Unit 10 | BitBtn1 | Переход на предыдущую форму | |
| 59 procedure TClown.FormCreate(Sender: TObject); | Unit 10 | Clown | Загрузка шрифта | |
| 60 procedure TClown.FormDestroy(Sender: TObject); | Unit 10 | Clown | Удаление шрифта | |
| 61 procedure TLoyalty.BitBtn1Click(Sender: TObject); | Unit 11 | BitBtn1 | Переход на предыдущую форму | |
| 62 procedure TLoyalty.FormCreate(Sender: TObject); | Unit 11 | Loyalty | Загрузка шрифта | |
| 63 procedure TLoyalty.FormDestroy(Sender: TObject); | Unit 11 | Loyalty | Удаление шрифта | |
| 64 procedure TLighthouse.BitBtn1Click(Sender: TObject); | Unit 12 | BitBtn1 | Переход на предыдущую форму | |
| 65 procedure TLighthouse.FormCreate(Sender: TObject); | Unit 12 | Lighthouse | Загрузка шрифта | |
| Продолжение таблицы 1 | | | | |
| 66 procedure TLighthouse.FormDestroy(Sender: TObject); | Unit 12 | Lighthouse | Удаление шрифта | |
| 67 procedure TRook.BitBtn1Click(Sender: TObject); | Unit 13 | BitBtn1 | Переход на предыдущую форму | |
| 68 procedure TRook.FormCreate(Sender: TObject); | Unit 13 | Rook | Загрузка шрифта | |
| 69 procedure TRook.FormDestroy(Sender: TObject); | Unit 13 | Rook | Удаление шрифта | |
| 70 procedure TWorker.BitBtn1Click(Sender: TObject); | Unit 14 | BitBtn1 | Переход на предыдущую форму | |
| 71 procedure TWorker.FormCreate(Sender: TObject); | Uni14 | Worker | Загрузка шрифта | |
| 72 procedure TWorker.FormDestroy(Sender: TObject); | Unit14 | Worker | Удаление шрифта | |
| 73 procedure TGrunwald.BitBtn1Click(Sender: TObject); | Unit 15 | BitBtn1 | Переход на предыдущую форму | |
| 74 procedure TGrunwald.FormCreate(Sender: TObject); | Unit15 | Grunwald | Загрузка шрифта | |
| 75 procedure TGrunwald.FormDestroy(Sender: TObject); | Unit 15 | Grunwald | Удаление шрифта | |
| 76 procedure TtTest.BitBtn1Click(Sender: TObject); | Test | BitBtn1 | Запуск теста | |
| 77 procedure TtTest.BitBtn2Click(Sender: TObject); | Test | BitBtn2 | Переход на предыдущую форму | |
| 78 procedure TtTest.FormCreate(Sender: TObject); | Test | tTest | Загрузка шрифта | |
| 79 procedure TtTest.FormDestroy(Sender: TObject); | Test | tTest | Удаление шрифта | |
| 80 procedure TtTest.RadioGroup1Click(Sender: TObject); | Test | RadioGroup1 | Загрузка теста | |
| 81 procedure TtTest.BitBtn2Click(Sender: TObject); | Test | BitBtn2 | Переход на предыдущую форму | |
| 82 procedure TtTest.FormCreate(Sender: TObject); | Test | tTest | Загрузка шрифта | |
| 83 procedure TtTest.FormDestroy(Sender: TObject); | Test | tTest | Удаление шрифта | |
| 84 procedure TtTest.RadioGroup1Click(Sender: TObject); | Test | RadioGroup1 | Загрузка теста | |

**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 | mMainMenu | Используется для создания справки проекта |
| TImage | zZastavka, mMainMenu, lLoadingR, lLoadingL, Form5, Zhiliber, Dialogue, Frog, Mail, Clown, Loyalty, Lighthouse, Rook, Worker, Grunwald, tTest | Используется как фон, изображение скульптуры, загрузка заставки и между формами |
| TTimer | zZastavka, lLoadingR, lLoadingL | Используется для определения длительности загрузки |
| TLabel | zZastavka, mMainMenu, lLoadingR, lLoadingL, Form5, Zhiliber, Dialogue, Frog, Mail, Clown, Loyalty, Lighthouse, Rook, Worker, Grunwald, tTest | Отображение надписей на форме |
| TBitBtn | mMainMenu, Form5, Zhiliber, Dialogue, Frog, Mail, Clown, Loyalty, Lighthouse, Rook, Worker, Grunwald, tTest | Используется для взаимодействия  пользователя с программой: для начала теста, закрытия программы, перехода на другую формы и т.д. |
| TRadioGroup | tTest | Используется для выбора варианта ответа в тесте |

**3.2 Спецификация программы**

Точное название проекта и его состав приводится в таблице 3.

Таблица 3 – Спецификация программы

|  |  |
| --- | --- |
| Имя файла | Назначение |
| 1 | 2 |
| USG.exe | Исполняемый файл проекта, используется для запуска программы на выполнение. |
| USG.dproj | Файл проекта, связывает все файлы, из которых состоит приложение. |
| Zastavka.pas | Файл программного модуля заставки |
| MainMenu.pas | Файл программного модуля главного меню |
| LoadingR.pas | Файл программного модуля загрузки на карту |
| LoadingL.pas | Файл программного модуля загрузки в главное меню |
| Unit5.pas | Файл программного модуля карты |
| Unit6.pas | Файл программного модуля скульптуры «Горожанин 18 века» |
| Unit7.pas | Файл программного модуля скульптуры «Диалог» |
| Unit8.pas | Файл программного модуля скульптуры «Лягушка-путешественница» |
| Unit9.pas | Файл программного модуля скульптуры «Почтовый ящик» |
| Unit10.pas | Файл программного модуля скульптуры «Петрушка» |
| Unit11.pas | Файл программного модуля скульптуры «Верность» |
| Unit12.pas | Файл программного модуля скульптуры «Колонна-маяк» |
| Unit13.pas | Файл программного модуля скульптуры «Ладья-нассад» |
| Unit14.pas | Файл программного модуля скульптуры «Дорожный рабочий» |
| Unit15.pas | Файл программного модуля скульптуры «Погоня на Грюнвальд» |
| Test.pas | Файл программного модуля теста |
| Zastavka.dfm | Форма загрузки |
| MainMenu.dfm | Форма главного меню |
| LoadingR.dfm | Форма с загрузкой на карту |
| LoadingL.dfm | Форма с загрузкой в главное меню |
| Unit5.dfm | Форма с картой |
| Unit6.dfm | Форма со скульптурой «Горожанин 18 века» |
| Unit7.dfm | Форма со скульптурой «Диалог» |
| Unit8.dfm | Форма со скульптурой «Лягушка-путешественница» |
| Unit9.dfm | Форма со скульптурой «Почтовый ящик» |
| Unit10.dfm | Форма со скульптурой «Петрушка» |
| Unit11.dfm | Форма со скульптурой «Верность» |
| Unit12.dfm | Форма со скульптурой «Колонна-маяк» |
| Unit13.dfm | Форма со скульптурой «Ладья-нассад» |
| Unit13.dfm | Форма со скульптурой «Погоня на Грюнвальд» |
| Test.dfm | Форма с тестом |
| test.txt | Тест |
| Продолжение таблицы 3 | |
| Setup.exe | Установочный пакет программы |
| 1.bmp | Изображение |
| 2.bmp | Изображение |
| 3.bmp | Изображение |
| 4.bmp | Изображение |
| 5.bmp | Изображение |
| 6.bmp | Изображение |
| 7.jpg | Изображение |
| 8.jpg | Изображение |
| 9.jpg | Изображение |
| 10.jpg | Изображение |
| 11.jpg | Изображение |
| 12.jpg | Изображение |
| 13.jpg | Изображение |
| 14.jpg | Изображение |
| 15.jpg | Изображение |
| 16.jpg | Изображение |
| 17.jpg | Изображение |
| 18.jpg | Изображение |
| 19.jpg | Изображение |
| 20.jpg | Изображение |
| 21.jpg | Изображение |
| 22.jpg | Изображение |
| 23.jpg | Изображение |
| 24.jpg | Изображение |
| 25.jpg | Изображение |
| 26.jpg | Изображение |
| 27.jpg | Изображение |
| 28.jpg | Изображение |

**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 запуска заставки | Запуск заставки | Запуск заставки | Выполнено |
| 2 | Проверка кнопки  «Начать» | Открытие формы с  картой | Открытие формы с  картой | Выполнено |
| 3 | Проверка кнопки-выход | Закрытие проекта | Закрытие проекта | Выполнено |
| 4 | Проверка перехода на форму с тестом | Переход на форму с тестом | Переход на форму с тестом | Выполнено |
| 5 | Проверка кнопки «Назад» | Переход на форму с главным меню | Переход на форму с главным меню | Выполнено |
| 6 | Проверка кнопки «Тест» | Открытие формы с тестом | Открытие формы с тестом | Выполнено |
| 7 | Проверка кнопки «Далее» | Переход на следующий вопрос в тесте | Переход на следующий вопрос в тесте | Выполнено |
| 8 | Проверка кнопки метки | Появление текста с названием скульптуры | Появление текста с названием скульптуры | Выполнено |
| 9 | Проверка кнопки перехода на скульптуру | Открытие формы с информацией о скульптуре | Открытие формы с информацией о скульптуре | Выполнено |
| 10 | Проверка кнопки  «На главное меню» | Переход на форму главного меню | Переход на форму главного меню | Выполнено |
| 12 | Проверка кнопки «Да» | Запуск теста | Запуск теста | Выполнено |

При разработке программного продукта было решено множество проблем, например, не осуществлялся переход к следующему изображению по нажатию стрелки или открывался доступ к последнему квесту, когда был пройден только первый, 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**

Разработать мобильное приложение, которое предоставляет пользователям доступ к информации о городских скульптурах и памятниках Гродно. Приложение должно включать интерактивную карту, текстовые и мультимедийные описания, историю объектов и образовательные тесты. Основной целевой аудиторией являются туристы, жители города и студенты. Тестирование проводилось на разных классах ЭВМ и работать с данной программой было комфортно. Программа разработана на ПК со следующими характеристиками:

* Процессор Intel Pentium N3710
* объем ОЗУ 4 Гб;
* видеокарта 910M
* операционная система Windows 10.

**5.2 Инсталляция**

Для того, чтобы установить программу необходимо запустить файл setup.exe. Появится окно установки приложения «USG».

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 «USG»

Вторым способом является запуск из каталога, в который устанавливалось приложение (по умолчанию C:\Program Files (x86)\USG

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**

После запуска приложения на экране нас встречает заставка на рисунке 3. После заставки нас будет встречать главное меню на рисунке 4.



Рисунок 2 – Логотип

Рисунок 3 – Заставка

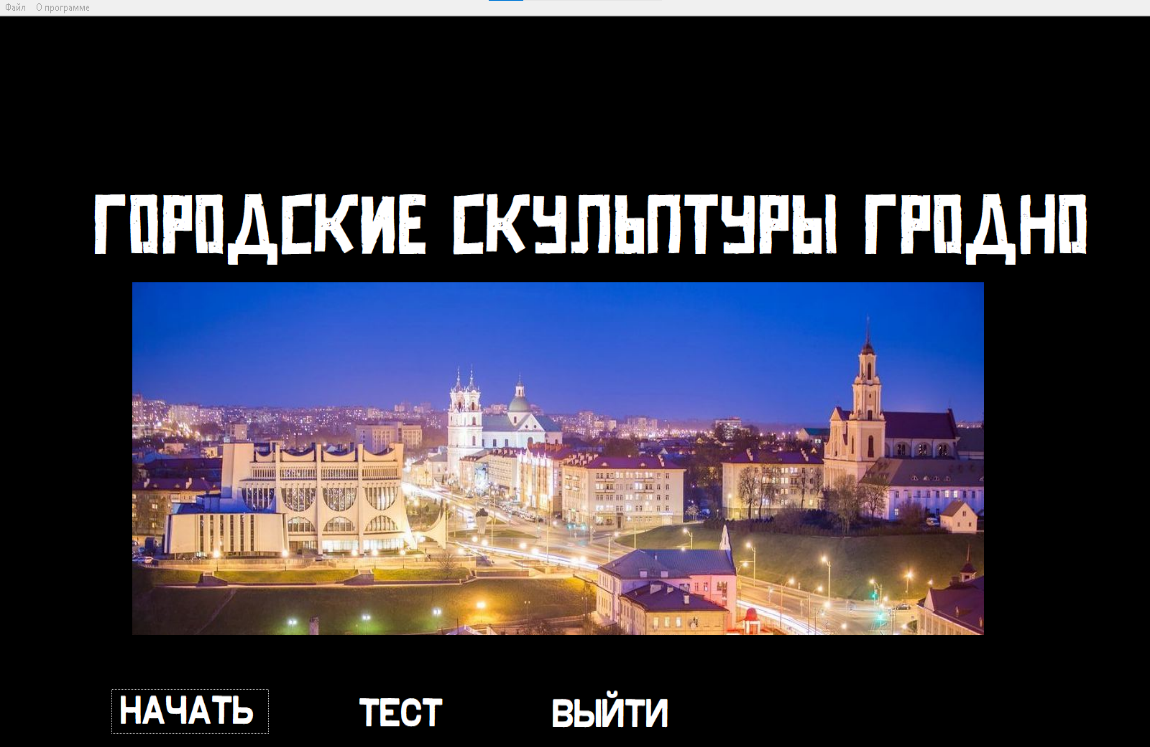


Рисунок 4 – Главное меню

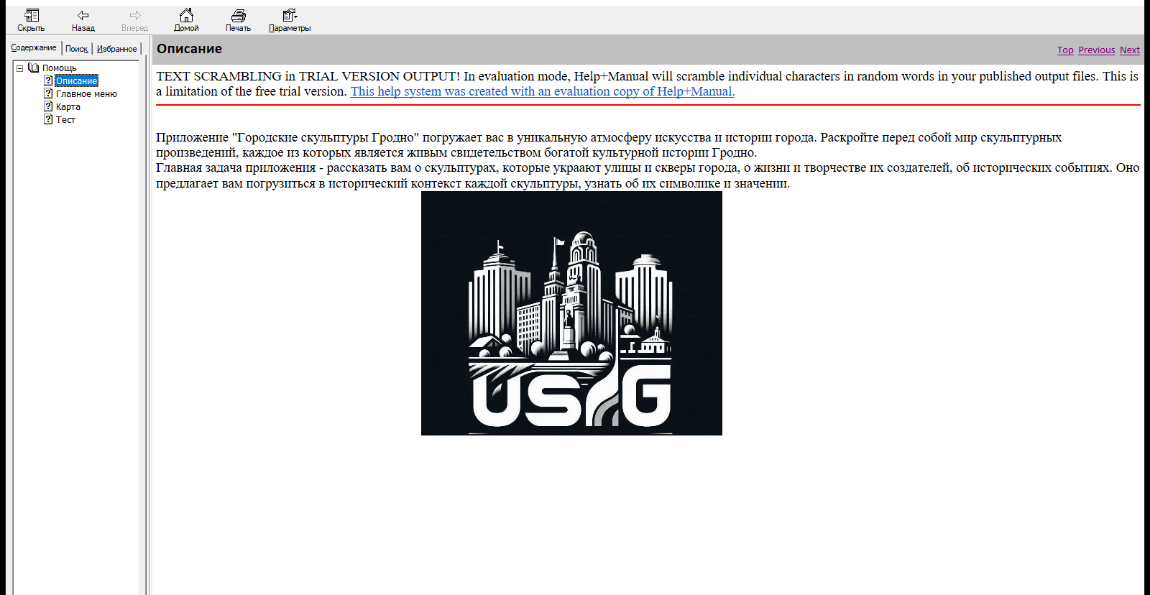
При нажатии на кнопку «Справку» будет осуществлен переход на справку с необходимой информацией, изображено на рисунке 5.

Рисунок 5 – Справка

При нажатии на кнопку «Начать» будет осуществлен переход на форму с картой, представлен на рисунке 6, где представлены метки и названия скульптур.



Рисунок 6 – Карта Гродно

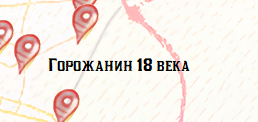
По нажатию на метку на карте будет появляться название скульптуры, изображенной на рисунке 7.

Рисунок 7 – Название скульптуры

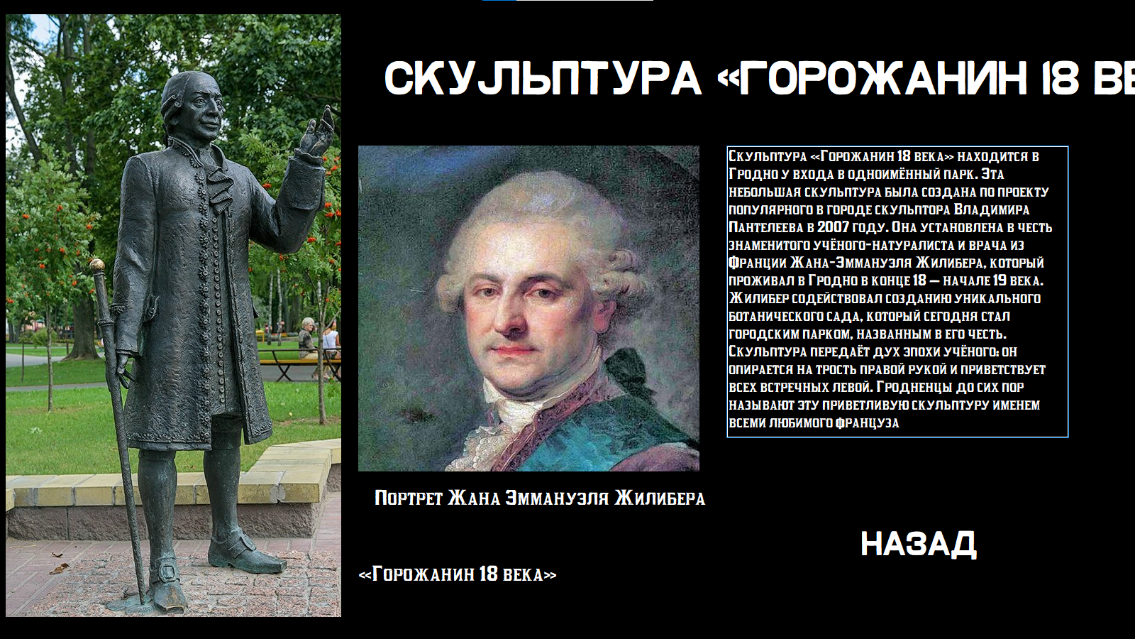
По нажатию на название скульптуры, произойдет переход на интересующую нас форму, представленной на рисунке 8.

Рисунок 8 – Форма с информацией и изображением скульптуры

По нажатию на кнопку «Тест» можно перейти на форму с тестом, представленный на рисунке 9.

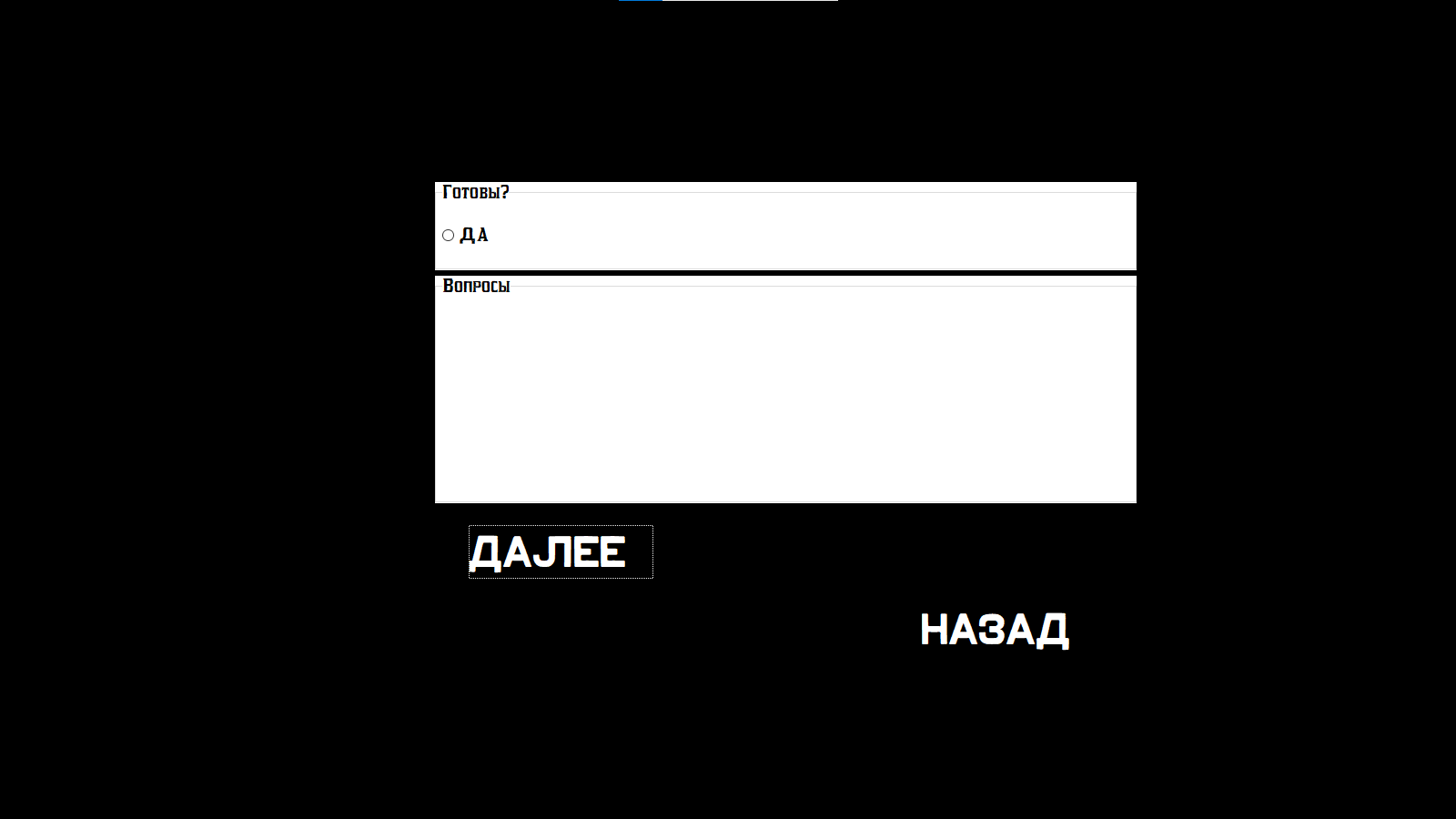
****

Рисунок 9 – Форма с тестом

**Заключение**

Задачей данного проекта, являлась разработка приложения, которое предоставляет пользователям доступ к информации о городских скульптурах Гродно.

В рамках выполнения поставленной задачи был разработан интуитивно понятный пользовательский интерфейс, обеспечивающий легкость и удобство взаимодействия. Информация о городских скульптурах была представлена в краткой и понятной форме, что позволяет пользователям быстро находить и усваивать необходимые данные.

Средой разработки является Embarcadero Delphi, выбранная за ее мощные возможности для создания кроссплатформенных приложений. Языком программирования является Object Pascal, благодаря его высокой производительности и широкому набору библиотек.

Приложение успешно предоставляет доступ к информации о скульптурах Гродно. Карта позволяет пользователям легко находить объекты, а тест добавляет развлекательные и образовательные элементы.

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

Приложение «Городские скульптуры Гродно» успешно внедрило современные технологии для повышения доступности и привлекательности информации о культурном наследии города. Разработка интуитивно понятного интерфейса и использование интерактивных инструментов позволили создать удобное и увлекательное средство для изучения городских скульптур, способствующее росту интереса и знаний пользователей о истории и культуре Гродно.

В целом, разработка и внедрение приложения "Городские скульптуры Гродно" демонстрирует, как современные технологии могут быть использованы для сохранения и популяризации культурного наследия, делая его доступным и интересным для всех категорий пользователей.

**Список использованных источников**

1. Сайт скульптур Гродно: [https://grodno.in/showplace/f:sculpture/. Дата](https://grodno.in/showplace/f:sculpture/.%20Дата) доступа: 15.06.2024.
2. Sputnik Беларусь: <https://sputnik.by/20150904/skulptury-grodno-1017037956.html>. Дата доступа: 15.06.2024
3. Симонович С.В., Евсеев Г.А., Алексеев А.Г. Специальная информатика: Учебное пособие. - М.: АСТ-ПРЕСС: Инфорком-Пресс, 2001. - 480 с. Дата доступа: 20.06.2024
4. Архангельский, А. Я. Delphi 7. Справочное пособие. - Москва: Бином-Пресс, 2014. - 1024 с. Дата доступа: 20.06.2024
5. Вощинская, Г.Э. Разработка компонентов в DELPHI. - Воронеж: ИПЦ ВГУ, 2007. - 57 с. Дата доступа: 20.06.2024

**Приложение А**

Листинг программы

unit Zastavka;

interface

uses

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

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

Vcl.Imaging.GIFImg;

type

TzZastavka = class(TForm)

Label1: TLabel;

Image1: TImage;

Timer1: TTimer;

Image3: TImage;

Image2: TImage;

procedure Timer1Timer(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure FormDestroy(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

zZastavka: TzZastavka;

c:integer;

fontway:string;

implementation

{$R \*.dfm}

uses MainMenu;

procedure TzZastavka.FormCreate(Sender: TObject);

begin

randomize;

(Image3.Picture.Graphic as TGIFImage).Animate := True;

c := Random(100);

if c < 50 then

begin

(Image1.Picture.Graphic as TGIFImage).Animate := True;

Image1.Visible := True;

end

else

begin

(Image2.Picture.Graphic as TGIFImage).Animate := True;

Image2.Visible := True;

end;

fontway:=extractfilepath(application.ExeName);

addfontresource(PChar(fontway + 'font\Redemption.ttf'));

end;

procedure TzZastavka.FormDestroy(Sender: TObject);

begin

fontway:=extractfilepath(application.ExeName);

removefontresource(PChar(fontway + 'Redemption\days.ttf'));

end;

procedure TzZastavka.Timer1Timer(Sender: TObject);

begin

Timer1.Destroy;

mMainMenu.Show;

zZastavka.Hide;

end;

end.

unit MainMenu;

interface

uses

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

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

Vcl.MPlayer, Vcl.Imaging.GIFImg, Vcl.Imaging.jpeg, Vcl.Menus, ShellAPI;

type

TmMainMenu = class(TForm)

Label1: TLabel;

BitBtn1: TBitBtn;

BitBtn3: TBitBtn;

BitBtn4: TBitBtn;

Image1: TImage;

MainMenu1: TMainMenu;

N1: TMenuItem;

N2: TMenuItem;

N3: TMenuItem;

N4: TMenuItem;

procedure BitBtn4Click(Sender: TObject);

procedure BitBtn1Click(Sender: TObject);

procedure BitBtn3Click(Sender: TObject);

procedure N2Click(Sender: TObject);

procedure N3Click(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure FormDestroy(Sender: TObject);

procedure N4Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

mMainMenu: TmMainMenu;

fontway:string;

implementation

{$R \*.dfm}

uses LoadingR, Test;

procedure TmMainMenu.BitBtn1Click(Sender: TObject);

begin

mMainMenu.Hide;

lLoadingR.Show;

end;

procedure TmMainMenu.BitBtn3Click(Sender: TObject);

begin

if MessageDlgPos('Перед тестом рекомендуется изучить карту.Вы уверены?',

mtInformation,[mbYes,mbNo],0,770,538)

=mrYes

then

mMainMenu.Hide;

tTest.Show;

end;

procedure TmMainMenu.BitBtn4Click(Sender: TObject);

begin

close;

end;

procedure TmMainMenu.FormCreate(Sender: TObject);

begin

fontway:=extractfilepath(application.ExeName);

addfontresource(PChar(fontway + 'font\Redemption.ttf'));

end;

procedure TmMainMenu.FormDestroy(Sender: TObject);

begin

fontway:=extractfilepath(application.ExeName);

removefontresource(PChar(fontway + 'font\Redemption.ttf'));

end;

procedure TmMainMenu.N2Click(Sender: TObject);

begin

close;

end;

procedure TmMainMenu.N3Click(Sender: TObject);

begin

Showmessage('Программу разработал Ганисевский Александр ПЗТ-41');

end;

procedure TmMainMenu.N4Click(Sender: TObject);

begin

ShellExecute(0,PChar('Open'),PChar('USGhelp.chm'), nil,nil,SW\_Show);

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.StdCtrls, Vcl.Imaging.GIFImg, Vcl.Buttons,

Vcl.ExtCtrls, Vcl.Imaging.pngimage;

type

TForm5 = class(TForm)

BitBtn1: TBitBtn;

Image0: TImage;

Image2: TImage;

Label1: TLabel;

Image3: TImage;

Label2: TLabel;

Image1: TImage;

Label3: TLabel;

Image4: TImage;

Label4: TLabel;

Image5: TImage;

Label5: TLabel;

Image6: TImage;

Label6: TLabel;

Image7: TImage;

Label7: TLabel;

Image8: TImage;

Label8: TLabel;

Image9: TImage;

Label9: TLabel;

Image10: TImage;

Label10: TLabel;

BitBtn2: TBitBtn;

procedure BitBtn1Click(Sender: TObject);

procedure Label1Click(Sender: TObject);

procedure Image2Click(Sender: TObject);

procedure Image3Click(Sender: TObject);

procedure Image1Click(Sender: TObject);

procedure Image4Click(Sender: TObject);

procedure Image6Click(Sender: TObject);

procedure Image7Click(Sender: TObject);

procedure Image8Click(Sender: TObject);

procedure Image9Click(Sender: TObject);

procedure Label3Click(Sender: TObject);

procedure Label8Click(Sender: TObject);

procedure Label7Click(Sender: TObject);

procedure Label2Click(Sender: TObject);

procedure Label9Click(Sender: TObject);

procedure Label4Click(Sender: TObject);

procedure Label6Click(Sender: TObject);

procedure Label5Click(Sender: TObject);

procedure Image5Click(Sender: TObject);

procedure Image10Click(Sender: TObject);

procedure Label10Click(Sender: TObject);

procedure BitBtn2Click(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure FormDestroy(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form5: TForm5;

fontway:string;

implementation

{$R \*.dfm}

uses LoadingL, Unit6, Unit10, Unit7, Unit12, Unit14, Unit13, Unit9, Unit8, Unit11,

Unit15, Test;

procedure TForm5.BitBtn1Click(Sender: TObject);

begin

lLoadingL.Show;

Form5.Hide;

end;

procedure TForm5.Image2Click(Sender: TObject);

begin

if Label2.Visible then

Label2.Visible := False

else

Label2.Visible := True;

end;

procedure TForm5.Image3Click(Sender: TObject);

begin

if Label3.Visible then

Label3.Visible := False

else

Label3.Visible := True;

end;

procedure TForm5.Image4Click(Sender: TObject);

begin

if Label4.Visible then

Label4.Visible := False

else

Label4.Visible := True;

end;

procedure TForm5.Image5Click(Sender: TObject);

begin

if Label5.Visible then

Label5.Visible := False

else

Label5.Visible := True;

end;

procedure TForm5.Image6Click(Sender: TObject);

begin

if Label6.Visible then

Label6.Visible := False

else

Label6.Visible := True;

end;

procedure TForm5.Image7Click(Sender: TObject);

begin

if Label7.Visible then

Label7.Visible := False

else

Label7.Visible := True;

end;

procedure TForm5.Image8Click(Sender: TObject);

begin

if Label8.Visible then

Label8.Visible := False

else

Label8.Visible := True;

end;

procedure TForm5.Image9Click(Sender: TObject);

begin

if Label9.Visible then

Label9.Visible := False

else

Label9.Visible := True;

end;

procedure TForm5.BitBtn2Click(Sender: TObject);

begin

Form5.Hide;

tTest.Show;

end;

procedure TForm5.FormCreate(Sender: TObject);

begin

fontway:=extractfilepath(application.ExeName);

addfontresource(PChar(fontway + 'font\RDRLino-Regular.ttf'));

end;

procedure TForm5.FormDestroy(Sender: TObject);

begin

fontway:=extractfilepath(application.ExeName);

removefontresource(PChar(fontway + 'font\RDRLino-Regular.ttf'));

end;

procedure TForm5.Image10Click(Sender: TObject);

begin

if Label10.Visible then

Label10.Visible := False

else

Label10.Visible := True;

end;

procedure TForm5.Image1Click(Sender: TObject);

begin

if Label1.Visible then

Label1.Visble := False

elsу

Label1.Visible := True;

end;

procedure TForm5.Label10Click(Sender: TObject);

begin

Form5.Hide;

Grunwald.Show;

end;

procedure TForm5.Label1Click(Sender: TObject);

begin

Form5.Hide;

Zhiliber.Show;

end;

procedure TForm5.Label2Click(Sender: TObject);

begin

Form5.Hide;

Clown.Show;

end;

procedure TForm5.Label3Click(Sender: TObject);

begin

Form5.Hide;

Dialogue.Show;

end;

procedure TForm5.Label4Click(Sender: TObject);

begin

Form5.Hide;

Lighthouse.Show;

end;

procedure TForm5.Label5Click(Sender: TObject);

begin

Form5.Hide;

Worker.Show;

end;

procedure TForm5.Label6Click(Sender: TObject);

begin

Form5.Hide;

Rook.Show;

end;

procedure TForm5.Label7Click(Sender: TObject);

begin

Form5.Hide;

Mail.Show;

end;

procedure TForm5.Label8Click(Sender: TObject);

begin

Form5.Hide;

Frog.Show;

end;

procedure TForm5.Label9Click(Sender: TObject);

begin

Form5.Hide;

Loyalty.Show;

end;

end.

unit LoadingR;

interface

uses

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

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

Vcl.Imaging.GIFImg;

type

TlLoadingR = class(TForm)

Label2: TLabel;

Timer2: TTimer;

Image1: TImage;

Image2: TImage;

Image3: TImage;

procedure Timer2Timer(Sender: TObject);

procedure FormShow(Sender: TObject);

procedure FormHide(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure FormDestroy(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

lLoadingR: TlLoadingR;

c: integer;

fontway:string;

implementation

{$R \*.dfm}

uses LoadingL, Unit5;

procedure TlLoadingR.FormCreate(Sender: TObject);

begin

fontway:=extractfilepath(application.ExeName);

addfontresource(PChar(fontway + 'font\RDRGothica-Regular.ttf'));

end;

procedure TlLoadingR.FormDestroy(Sender: TObject);

begin

fontway:=extractfilepath(application.ExeName);

removefontresource(PChar(fontway + 'font\RDRGothica-Regular.ttf'));

end;

procedure TlLoadingR.FormHide(Sender: TObject);

begin

Timer2.Enabled := False;

end;

procedure TlLoadingR.FormShow(Sender: TObject);

begin

Randomize;

c := Random(150);

Image1.Visible := False;

Image2.Visible := False;

Image3.Visible := False;

if Image1.Picture.Graphic is TGIFImage then

(Image1.Picture.Graphic as TGIFImage).Animate := False;

if Image2.Picture.Graphic is TGIFImage then

(Image2.Picture.Graphic as TGIFImage).Animate := False;

if Image3.Picture.Graphic is TGIFImage then

(Image3.Picture.Graphic as TGIFImage).Animate := False;

if c < 50 then

begin

Image1.Visible := True;

(Image1.Picture.Graphic as TGIFImage).Animate := True;

end

else if c < 100 then

begin

Image2.Visible := True;

(Image2.Picture.Graphic as TGIFImage).Animate := True;

end

else

begin

Image3.Visible := True;

(Image3.Picture.Graphic as TGIFImage).Animate := True;

end;

Timer2.Enabled := True;

end;

procedure TlLoadingR.Timer2Timer(Sender: TObject);

begin

Form5.Show;

lLoadingR.Hide;

end;

end.

unit LoadingL;

interface

uses

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

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.StdCtrls, Vcl.Buttons,

Vcl.Imaging.GIFImg, Vcl.ExtCtrls;

type

TlLoadingL = class(TForm)

Image1: TImage;

Label1: TLabel;

Image2: TImage;

Image3: TImage;

Timer3: TTimer;

procedure FormShow(Sender: TObject);

procedure Timer3Timer(Sender: TObject);

procedure FormHide(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure FormDestroy(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

lLoadingL: TlLoadingL;

c: integer;

fontway:string;

implementation

{$R \*.dfm}

uses MainMenu;

procedure TlLoadingL.FormCreate(Sender: TObject);

begin

fontway:=extractfilepath(application.ExeName);

addfontresource(PChar(fontway + 'font\RDRGothica-Regular.ttf'));

end;

procedure TlLoadingL.FormDestroy(Sender: TObject);

begin

fontway:=extractfilepath(application.ExeName);

removefontresource(PChar(fontway + 'font\RDRGothica-Regular.ttf'));

end;

procedure TlLoadingL.FormHide(Sender: TObject);

begin

Timer3.Enabled := False;

end;

procedure TlLoadingL.FormShow(Sender: TObject);

begin

Randomize;

c := Random(150);

Image1.Visible := False;

Image2.Visible := False;

Image3.Visible := False;

if Image1.Picture.Graphic is TGIFImage then

(Image1.Picture.Graphic as TGIFImage).Animate := False;

if Image2.Picture.Graphic is TGIFImage then

(Image2.Picture.Graphic as TGIFImage).Animate := False;

if Image3.Picture.Graphic is TGIFImage then

(Image3.Picture.Graphic as TGIFImage).Animate := False;

if c < 50 then

begin

Image1.Visible := True;

(Image1.Picture.Graphic as TGIFImage).Animate := True;

end

else if c < 100 then

begin

Image2.Visible := True;

(Image2.Picture.Graphic as TGIFImage).Animate := True;

end

else

begin

Image3.Visible := True;

(Image3.Picture.Graphic as TGIFImage).Animate := True;

end;

Timer3.Enabled := True;

end;

procedure TlLoadingL.Timer3Timer(Sender: TObject);

begin

mMainMenu.Show;

lLoadingL.Hide;

end;

end.

unit Unit6;

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, Vcl.Buttons;

type

TZhiliber = class(TForm)

Label1: TLabel;

Image1: TImage;

Label2: TLabel;

Memo1: TMemo;

Image2: TImage;

Label3: TLabel;

BitBtn1: TBitBtn;

procedure BitBtn1Click(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure FormDestroy(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Zhiliber: TZhiliber;

fontway:string;

implementation

{$R \*.dfm}

uses Unit5;

procedure TZhiliber.BitBtn1Click(Sender: TObject);

begin

Form5.Show;

Zhiliber.Hide;

end;

procedure TZhiliber.FormCreate(Sender: TObject);

begin

fontway:=extractfilepath(application.ExeName);

addfontresource(PChar(fontway + 'font\RDRGothica-Regular.ttf'));

end;

procedure TZhiliber.FormDestroy(Sender: TObject);

begin

fontway:=extractfilepath(application.ExeName);

removefontresource(PChar(fontway + 'font\RDRGothica-Regular.ttf'));

end;

end.

unit Unit7;

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, Vcl.Buttons;

type

TDialogue = class(TForm)

Label1: TLabel;

Image1: TImage;

Memo1: TMemo;

Image2: TImage;

BitBtn1: TBitBtn;

procedure BitBtn1Click(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure FormDestroy(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Dialogue: TDialogue;

fontway:string;

implementation

{$R \*.dfm}

uses Unit5;

procedure TDialogue.BitBtn1Click(Sender: TObject);

begin

Form5.Show;

Dialogue.Hide;

end;

procedure TDialogue.FormCreate(Sender: TObject);

begin

fontway:=extractfilepath(application.ExeName);

addfontresource(PChar(fontway + 'font\RDRGothica-Regular.ttf'));

end;

procedure TDialogue.FormDestroy(Sender: TObject);

begin

fontway:=extractfilepath(application.ExeName);

removefontresource(PChar(fontway + 'font\RDRGothica-Regular.ttf'));

end;

end.

unit Unit8;

interface

uses

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

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

Vcl.Imaging.jpeg, Vcl.Buttons;

type

TFrog = class(TForm)

Label1: TLabel;

Image1: TImage;

Image2: TImage;

Memo1: TMemo;

BitBtn1: TBitBtn;

procedure BitBtn1Click(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure FormDestroy(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Frog: TFrog;

fontway:string;

implementation

{$R \*.dfm}

uses Unit5;

procedure TFrog.BitBtn1Click(Sender: TObject);

begin

Form5.Show;

Frog.Hide;

end;

procedure TFrog.FormCreate(Sender: TObject);

begin

fontway:=extractfilepath(application.ExeName);

addfontresource(PChar(fontway + 'font\RDRGothica-Regular.ttf'));

end;

procedure TFrog.FormDestroy(Sender: TObject);

begin

fontway:=extractfilepath(application.ExeName);

removefontresource(PChar(fontway + 'font\RDRGothica-Regular.ttf'));

end;

end.

unit Unit9;

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, Vcl.Buttons;

type

TMail = class(TForm)

Label1: TLabel;

p: TImage;

Image2: TImage;

Memo1: TMemo;

Image1: TImage;

BitBtn1: TBitBtn;

procedure BitBtn1Click(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure FormDestroy(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Mail: TMail;

fontway:string;

implementation

{$R \*.dfm}

uses Unit5;

procedure TMail.BitBtn1Click(Sender: TObject);

begin

Form5.Show;

Mail.Hide;

end;

procedure TMail.FormCreate(Sender: TObject);

begin

fontway:=extractfilepath(application.ExeName);

addfontresource(PChar(fontway + 'font\RDRGothica-Regular.ttf'));

end;

procedure TMail.FormDestroy(Sender: TObject);

begin

fontway:=extractfilepath(application.ExeName);

removefontresource(PChar(fontway + 'font\RDRGothica-Regular.ttf'));

end;

end.

unit Unit10;

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, Vcl.Buttons;

type

TClown = class(TForm)

Label1: TLabel;

Memo1: TMemo;

Image1: TImage;

Image2: TImage;

Image3: TImage;

BitBtn1: TBitBtn;

procedure FormCreate(Sender: TObject);

procedure FormDestroy(Sender: TObject);

procedure BitBtn1Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Clown: TClown;

fontway:string;

implementation

{$R \*.dfm}

uses Unit5;

procedure TClown.BitBtn1Click(Sender: TObject);

begin

Form5.Show;

Clown.Hide;

end;

procedure TClown.FormCreate(Sender: TObject);

begin

fontway:=extractfilepath(application.ExeName);

addfontresource(PChar(fontway + 'font\RDRGothica-Regular.ttf'));

end;

procedure TClown.FormDestroy(Sender: TObject);

begin

fontway:=extractfilepath(application.ExeName);

removefontresource(PChar(fontway + 'font\RDRGothica-Regular.ttf'));

end;

end.

unit Unit11;

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.jpeg, Vcl.Buttons;

type

TLoyalty = class(TForm)

Label1: TLabel;

Memo1: TMemo;

Image1: TImage;

Image2: TImage;

BitBtn1: TBitBtn;

procedure BitBtn1Click(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure FormDestroy(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Loyalty: TLoyalty;

fontway:string;

implementation

{$R \*.dfm}

uses Unit5;

procedure TLoyalty.BitBtn1Click(Sender: TObject);

begin

Form5.Show;

Loyalty.Hide;

end;

procedure TLoyalty.FormCreate(Sender: TObject);

begin

fontway:=extractfilepath(application.ExeName);

addfontresource(PChar(fontway + 'font\RDRGothica-Regular.ttf'));

end;

procedure TLoyalty.FormDestroy(Sender: TObject);

begin

fontway:=extractfilepath(application.ExeName);

removefontresource(PChar(fontway + 'font\RDRGothica-Regular.ttf'));

end;

end.

unit Unit12;

interface

uses

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

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.StdCtrls, Vcl.Buttons,

Vcl.Imaging.jpeg, Vcl.ExtCtrls;

type

TLighthouse = class(TForm)

Label1: TLabel;

Memo1: TMemo;

Image1: TImage;

BitBtn1: TBitBtn;

procedure BitBtn1Click(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure FormDestroy(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Lighthouse: TLighthouse;

fontway:string;

implementation

{$R \*.dfm}

uses Unit5;

procedure TLighthouse.BitBtn1Click(Sender: TObject);

begin

Form5.Show;

Lighthouse.Hide;

end;

procedure TLighthouse.FormCreate(Sender: TObject);

begin

fontway:=extractfilepath(application.ExeName);

addfontresource(PChar(fontway + 'font\RDRGothica-Regular.ttf'));

end;

procedure TLighthouse.FormDestroy(Sender: TObject);

begin

fontway:=extractfilepath(application.ExeName);

removefontresource(PChar(fontway + 'font\RDRGothica-Regular.ttf'));

end;

end.

unit Unit13;

interface

uses

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

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

Vcl.Imaging.jpeg, Vcl.Buttons;

type

TRook = class(TForm)

Label1: TLabel;

Memo1: TMemo;

Image1: TImage;

BitBtn1: TBitBtn;

procedure BitBtn1Click(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure FormDestroy(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Rook: TRook;

fontway:string;

implementation

{$R \*.dfm}

uses Unit5;

procedure TRook.BitBtn1Click(Sender: TObject);

begin

Form5.Show;

Rook.Hide;

end;

procedure TRook.FormCreate(Sender: TObject);

begin

fontway:=extractfilepath(application.ExeName);

addfontresource(PChar(fontway + 'font\RDRGothica-Regular.ttf'));

end;

procedure TRook.FormDestroy(Sender: TObject);

begin

fontway:=extractfilepath(application.ExeName);

removefontresource(PChar(fontway + 'font\RDRGothica-Regular.ttf'));

end;

end.

unit Unit14;

interface

uses

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

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

Vcl.Imaging.jpeg;

type

TWorker = class(TForm)

Label1: TLabel;

BitBtn1: TBitBtn;

Image1: TImage;

Image2: TImage;

Memo1: TMemo;

procedure BitBtn1Click(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure FormDestroy(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Worker: TWorker;

fontway:string;

implementation

{$R \*.dfm}

uses Unit5;

procedure TWorker.BitBtn1Click(Sender: TObject);

begin

Form5.Show;

Worker.Hide;

end;

procedure TWorker.FormCreate(Sender: TObject);

begin

fontway:=extractfilepath(application.ExeName);

addfontresource(PChar(fontway + 'font\RDRGothica-Regular.ttf'));

end;

procedure TWorker.FormDestroy(Sender: TObject);

begin

fontway:=extractfilepath(application.ExeName);

removefontresource(PChar(fontway + 'font\RDRGothica-Regular.ttf'));

end;

end.

unit Unit15;

interface

uses

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

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

Vcl.Imaging.jpeg;

type

TGrunwald = class(TForm)

Image1: TImage;

Image2: TImage;

Label1: TLabel;

Memo1: TMemo;

BitBtn1: TBitBtn;

procedure BitBtn1Click(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure FormDestroy(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Grunwald: TGrunwald;

fontway:string;

implementation

{$R \*.dfm}

uses Unit5;

procedure TGrunwald.BitBtn1Click(Sender: TObject);

begin

Grunwald.Hide;

Form5.Show;

end;

procedure TGrunwald.FormCreate(Sender: TObject);

begin

fontway:=extractfilepath(application.ExeName);

addfontresource(PChar(fontway + 'font\RDRGothica-Regular.ttf'));

end;

procedure TGrunwald.FormDestroy(Sender: TObject);

begin

fontway:=extractfilepath(application.ExeName);

removefontresource(PChar(fontway + 'font\RDRGothica-Regular.ttf'));

end;

end.

unit Test;

interface

uses

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

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.StdCtrls, Vcl.ExtCtrls, Vcl.Buttons;

type

TtTest = class(TForm)

RadioGroup1: TRadioGroup;

RadioGroup2: TRadioGroup;

Label1: TLabel;

BitBtn1: TBitBtn;

BitBtn2: TBitBtn;

BitBtn3: TBitBtn;

procedure RadioGroup1Click(Sender: TObject);

procedure BitBtn1Click(Sender: TObject);

procedure BitBtn2Click(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure FormDestroy(Sender: TObject);

procedure BitBtn3Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

tTest: TtTest;

f: text;

fontway, s: string;

Nvern, ball, totalQuestions: integer;

implementation

{$R \*.dfm}

uses Unit5, MainMenu;

procedure TtTest.BitBtn1Click(Sender: TObject);

begin

if (RadioGroup2.ItemIndex > -1) and (not Eof(f)) then

begin

totalQuestions := totalQuestions + 1;

if RadioGroup2.ItemIndex = Nvern - 1 then

begin

ball := ball + 1;

end

else

begin

Label1.Caption := '';

end;

RadioGroup2.Items.Clear;

repeat

if (s[1] = '-') then

begin

delete(s, 1, 1);

RadioGroup2.Caption := s;

end

else if s[1] = '\*' then

begin

delete(s, 1, 1);

Nvern := StrToInt(s);

end

else

RadioGroup2.Items.Add(s);

readln(f, s);

until (s[1] = '-') or Eof(f);

end

else if Eof(f) then

begin

delete(s, 1, 1);

Nvern := StrToInt(s);

totalQuestions := totalQuestions + 1;

if RadioGroup2.ItemIndex = Nvern - 1 then

begin

ball := ball + 1;

end

else

Label1.Caption := Format('Ваш результат: %d%% правильных ответов', [Round(ball / totalQuestions \* 100)]);

CloseFile(f);

BitBtn1.Visible := False;

BitBtn2.Visible := True;

end;

end;

procedure TtTest.BitBtn2Click(Sender: TObject);

begin

mMainMenu.Show;

tTest.Hide;

end;

procedure TtTest.BitBtn3Click(Sender: TObject);

begin

Form5.Show;

tTest.Hide;

end;

procedure TtTest.FormCreate(Sender: TObject);

begin

fontway:=extractfilepath(application.ExeName);

addfontresource(PChar(fontway + 'font\RDRLino-Regular.ttf'));

end;

procedure TtTest.FormDestroy(Sender: TObject);

begin

fontway:=extractfilepath(application.ExeName);

removefontresource(PChar(fontway + 'font\RDRLino-Regular.ttf'));

end;

procedure TtTest.RadioGroup1Click(Sender: TObject);

begin

RadioGroup1.Enabled := false;

RadioGroup2.Enabled := true;

BitBtn1.Enabled := true;

case RadioGroup1.ItemIndex of

0: AssignFile(f, ExtractFilePath(Application.ExeName) + 'Test.txt');

end;

reset(f);

readln(f, s);

ball := 0;

totalQuestions := 0;

repeat

if (s[1] = '-') then

begin

delete(s, 1, 1);

RadioGroup2.Caption := s;

end

else if s[1] = '\*' then

begin

delete(s, 1, 1);

Nvern := StrToInt(s);

end

else

RadioGroup2.Items.Add(s);

readln(f, s);

until (s[1] = '-') or Eof(f);

end;

end.