**Министерство науки и высшего образования Российской Федерации**

**ФГАОУ ВО «УрФУ имени первого Президента России Б.Н. Ельцина»**

Кафедра «машиностроения (департамент)»

**Отчет по лабораторной работе № 2**

«API ОС Windows»

**Выполнил:**

**Крашенинников Ян**

**Группа: НМТ-313901**

**Преподаватель:**

**Истомин А.С.**

**Вариант 9:**

**• Фигура: Круг**

**• Местоположение: X: Посередине, Y: Сверху**

**• Цвет: Черный**

Для начала был создан файл в paint и сохранен в формате bmp:

Изображение выглядит как снимок экрана, Мультимедийное программное обеспечение, программное обеспечение, Графическое программное обеспечение

Автоматически созданное описание

Рис. 1 – создание картинки

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Рис. 2 – сохранение файла в формате bmp

Далее начинаем писать программу

Код представляет собой обработку сообщений в оконной процедуре (`WndProc`). Он отвечает за обработку различных событий и действий пользователя в приложении, включая открытие BMP-файла.

1. В начале кода импортируются необходимые библиотеки и определяются несколько глобальных переменных, включая `hBitmap` и `IDM\_OPEN`.

2. Далее идет объявление структуры `ImgInfo`, которая используется для хранения информации о пикселях в BMP-файле.

3. Функция `OpenBMPFile` открывает диалоговое окно выбора файла с помощью функции `GetOpenFileName`. Если пользователь выбирает BMP-файл, то путь к нему сохраняется в переменной `selectedFile`.

4. Функция `readPixelColorsFromFile` открывает выбранный BMP-файл, считывает его заголовок и информацию о пикселях. Она создает вектор `pixelColors`, в котором хранятся все цвета пикселей изображения. Затем она инициализирует структуру `ImgInfo`, заполняет ее данными о пикселях изображения и возвращает указатель на эту структуру.

5. Функция `drawShape` отрисовывает изображение на контексте устройства (`hdcWindow`). Она проходит в циклах по всем координатам пикселей изображения и использует функцию `SetPixel` для установки цветов пикселей.

6. Функция `WndProc` является обработчиком сообщений для окна. Внутри нее происходит обработка сообщения `WM\_COMMAND`. Если идентификатор команды соответствует `IDM\_OPEN`, то вызывается функция `OpenBMPFile` для открытия BMP-файла. Затем вызывается функция `readPixelColorsFromFile` для чтения содержимого файла и сохранения информации о пикселях в структуре `ImgInfo`. Если чтение файла прошло успешно, то вызывается функция `InvalidateRect` для пометки клиентской области окна как недействительной и вызова перерисовки окна.

7. В обработчике сообщения `WM\_PAINT` вызываются функции `BeginPaint` и `EndPaint` для начала и окончания рисования. Внутри обработчика вызывается функция `drawShape` для отображения изображения на контексте устройства `hdc`. Затем функция завершается возвращением значения 0.

8. Обработчик сообщения `WM\_SIZE`:

Это сообщение вызывается всякий раз, когда размер окна изменяется пользователем. В обработчике вызывается функция `InvalidateRect` соответствующая окна (указанное `hwnd`), чтобы пометить всю клиентскую область окна как недействительную. Это приводит к вызову перерисовки окна и обновлению его содержимого с учетом нового размера.

9. Обработчик сообщения `WM\_CLOSE`:

- Это сообщение возникает при нажатии пользователем на кнопку закрытия окна.

- В обработчике вызывается функция `DeleteObject`, чтобы удалить объект `hBitmap`. Предполагается, что `hBitmap` - это ресурс, связанный с окном.

- Затем вызывается функция `PostQuitMessage`, которая помещает сообщение выхода (`WM\_QUIT`) в очередь сообщений, чтобы завершить цикл обработки сообщений и закрыть приложение.

Кроме того, код в функции `WinMain` содержит следующее:

- В функии `CreateMenu` и `CreatePopupMenu` были добавлены вызовы, чтобы создать главное меню и подменю.

- В функции `AppendMenu` добавлен вызов, чтобы добавить элемент меню в подменю.

- Установка меню для окна выполняется с помощью вызова функции `SetMenu`, которой передается дескриптор окна (`hwnd`) и дескриптор созданного главного меню (`hMenu`).

- После отображения окна (`ShowWindow`), запускается цикл обработки сообщений (`GetMessage`, `TranslateMessage`, `DispatchMessage`).

В конце концов у нас при запуске получается следующее:

Изображение выглядит как снимок экрана, компьютер, дисплей, программное обеспечение

Автоматически созданное описание

Рис. 3 – Окно приложения

В этом окне также есть кнопка «файл», нажав на которую можно будет выбрать наш рисунок

Изображение выглядит как программное обеспечение, снимок экрана, текст, Мультимедийное программное обеспечение

Автоматически созданное описание

Рис. 4 – выбор рисунка

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Рис. 5 – нарисованный круг по варианту в центре сверху экрана

КРАТКАЯ ИДЕЯ МОЕГО ШЕДЕВРА:

Простыми словами вкратце: программа получает изображение, обрезает изображение ПО РАЗМЕРАМ ФИГУРЫ И УДАЛЯЕТ ФОН, ДЕЛАЕТ ЕГО БЕЛЫМ. То есть идея в том, что не просто фон делается белым, а именно создается новое изображение с измененными параметрами, размер картинки равен максимальным размерам фигуры

Затем в функции отрисовки весь экран сначала заливается белым цветом (чтобы не было черных артефактов как в моем первом варианте, там все плохо было) а потом вычисляется середина обрезанного изображения и ставится в нужное место на окне (даже если менять размеры окна), то есть пиксельная середина горизонтали окна ВСЕГДА равна пиксельной середине ширины фигуры, поэтому никакие масштабирования не страшны!!!

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Рис. 6 – красота №1

Изображение выглядит как текст, снимок экрана, логотип

Автоматически созданное описание

Рис.7 – красота №2 во весь экран

Вывод:

В целом, работой доволен, тяжелее чем ассемблер. После сдачи потратил еще неделю чтобы довести до совершенства (в моем понимании, в понимании того, что в теории ожидалось вами в данной работе) эйфория получена в полном объеме, спасибо!

В результате выполнения практической работы было создано оконное приложение для рабочих столов, которое отображает изображение из файла в заданном местоположении окна согласно варианту работы.

Полученные навыки и знания могут быть использованы для разработки других оконных приложений на платформе Windows. Также, работа с файлами и отображение графики в окне являются основными аспектами многих приложений, и теперь вы обладаете базовыми навыками в этих областях.

В дальнейшем можно расширить созданное приложение, добавив различные возможности.

Практическая работа позволила получить практический опыт работы с файлами и разработкой оконных приложений, что может быть полезным для решения различных задач и создания полноценных приложений на платформе Windows.

Ccылка: https://github.com/krosshiik/OSprac2