МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ автономное ОБРАЗОВАТЕЛЬНОЕ

УЧРЕЖДЕНИЕ ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ

МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

(ФГБОУ ВПО МПУ)



Кафедра СМАРТ-технологии

Практическая работа № 6

«Изучение возможностей языка C# в обработке и представлении графической информации»

По дисциплине «Программирование и алгоритмизация на языках высокого уровня»

Группа \_\_\_\_241-324 \_\_\_\_\_\_

№ группы

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Сальников Л.В.

Подпись студента

Дата \_\_\_27.05.2025\_\_\_\_

Дата сдачи

Преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Крыжановская Т.Г.

Подпись преподавателя

Преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Клецкин А.А.

Подпись преподавателя

2025

Цель:

• Создание приложения для отрисовки графических примитивов и текста;

• Создать приложение для работы с битовыми картами.

Выполнение задания:

**Этап 1: Создание проекта**

* Создал новый проект типа **Windows Forms App (.NET 8.0)**.
* Назвал, Drawer.

### Этап 2: Дизайн интерфейса

Разместил на форме следующие элементы:

|  |  |
| --- | --- |
| **Элемент управления** | Назначение |
| PictureBox | Для отображения рисунка |
| Button "Загрузить файл" | Для открытия файла для просмотра |
| Button "Сохранить PNG" | Для сохранения загруженного файла в формате PNG |

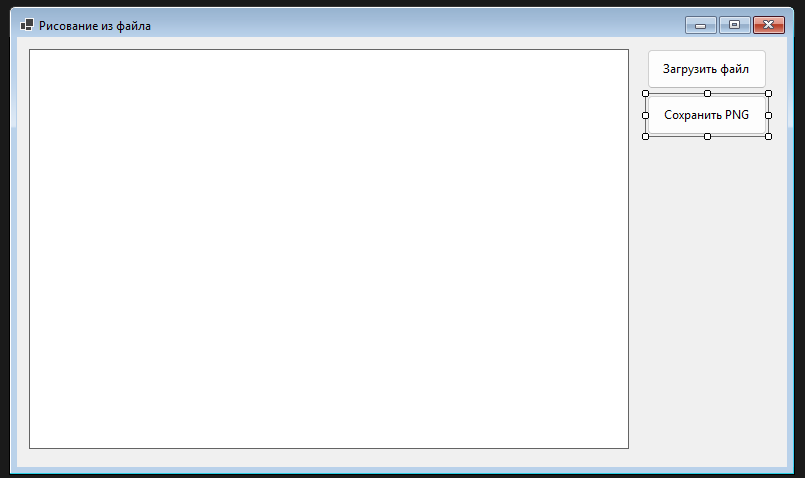


Рисунок 1

Этап 3: Подключение обработчиков

Подключил обработчики для события Click кнопки загрузки:

* Вызывается диалоговое окно выбора файла с фильтром «(\*.txt)|\*.txt»
* Очищается окно вывода
* Считываются все строки выбранного файла
* Построчно выполняются команды

Обработчик кнопки загрузки:

private void btnLoadFile\_Click(object sender, EventArgs e)

{

using (var openFileDialog = new OpenFileDialog())

{

openFileDialog.Filter = "Text Files (\*.txt)|\*.txt";

if (openFileDialog.ShowDialog() == DialogResult.OK)

{

// Переинициализируем поверхность для очистки

InitializeDrawingSurface();

graphics.Clear(Color.White);

try

{

ProcessFile(openFileDialog.FileName);

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка: {ex.Message}");

}

pictureBox.Invalidate();

}

}

Функция, обрабатывающая строки из файла

private void ProcessFile(string filePath)

{

var lines = File.ReadAllLines(filePath);

foreach (var line in lines)

{

var parts = line.Split(new[] { ' ' }, StringSplitOptions.RemoveEmptyEntries);

if (parts.Length == 0) continue;

var command = parts[0].ToUpper();

ProcessCommand(command, parts);

}

}

Этап 4: Обработка команд

Строки разделены на части, происходит последовательная проверка типа отрисовываемого элемента, если не было совпадений по типам, то ничего не происходит, иначе отрисовывается элемент того типа, который был задан по введенным параметрам.

private void ProcessCommand(string command, string[] parts)

{

switch (command)

{

case "TEXT":

if (parts.Length >= 4 &&

int.TryParse(parts[1], out int tx) &&

int.TryParse(parts[2], out int ty))

{

var text = string.Join(" ", parts, 3, parts.Length - 3);

graphics.DrawString(text, Font, Brushes.Black, tx, ty);

}

break;

case "CIRCLE":

if (parts.Length == 4 &&

int.TryParse(parts[1], out int cx) &&

int.TryParse(parts[2], out int cy) &&

int.TryParse(parts[3], out int radius))

{

using (var pen = new Pen(Color.Red))

{

graphics.DrawEllipse(pen, cx - radius, cy - radius, radius \* 2, radius \* 2);

}

}

break;

case "RECT":

if (parts.Length == 5 &&

int.TryParse(parts[1], out int rx) &&

int.TryParse(parts[2], out int ry) &&

int.TryParse(parts[3], out int rw) &&

int.TryParse(parts[4], out int rh))

{

using (var pen = new Pen(Color.Blue))

{

graphics.DrawRectangle(pen, rx, ry, rw, rh);

}

}

break;

case "LINE":

if (parts.Length == 5 &&

int.TryParse(parts[1], out int x1) &&

int.TryParse(parts[2], out int y1) &&

int.TryParse(parts[3], out int x2) &&

int.TryParse(parts[4], out int y2))

{

using (var pen = new Pen(Color.Green))

{

graphics.DrawLine(pen, x1, y1, x2, y2);

}

}

break;

case "TRIANGLE":

if (parts.Length == 7 &&

int.TryParse(parts[1], out int t1x) &&

int.TryParse(parts[2], out int t1y) &&

int.TryParse(parts[3], out int t2x) &&

int.TryParse(parts[4], out int t2y) &&

int.TryParse(parts[5], out int t3x) &&

int.TryParse(parts[6], out int t3y))

{

using (var pen = new Pen(Color.Purple))

{

var triangle = new[]

{

new Point(t1x, t1y),

new Point(t2x, t2y),

new Point(t3x, t3y)

};

graphics.DrawPolygon(pen, triangle);

}

}

break;

}

}

Этап 5. Обработка ошибок

Чтобы предотвратить непредвиденные ошибки в программе используется конструкция try-catch, если в теле try будет выявлена ошибка, то программа «поймает» её и корректно отработает.

Этап 6: Тестирование

Для проверки корректности выполнения задания, сделал файл, в котором содержатся команды для программы

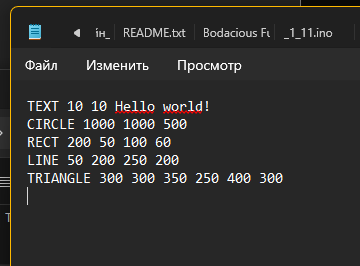


Рисунок 2 Команды, записанные в файл

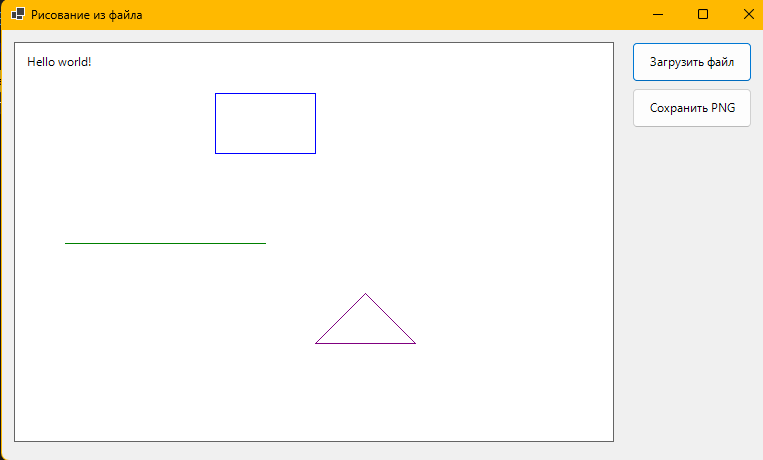


Рисунок 3 Результат выполнения программы

Вывод:

• Создал приложения для отрисовки графических примитивов и текста;

• Создал приложение для работы с битовыми картами.