Министерство транспорта Российской Федерации

Федеральное государственное автономное образовательное

учреждение высшего образования

«Российский университет транспорта»

(ФГАОУ ВО РУТ(МИИТ), РУТ (МИИТ)

Институт транспортной техники и систем управления

Кафедра «Управление и защита информации»

Лабораторная работа № 16

по дисциплине: «Программирование и основы алгоритмизации»

на тему: «Знакомство с режимом разработки *Windows Forms*»

Выполнил: ст. гр. ТУУ-111

Богомолов В.Н.

Вариант №1

01.06.2025

(дата выполнения)

Проверил: к.т.н., доц. Сафронов А.И.

07.06.2025

(дата приёмки)

Москва – 2025 г.

1. **Цель**

Разработать учебное приложение рабочего стола (*Desktop Application*) с простым графическим пользовательским интерфейсом (*GUI*), который может послужить примером для демонстрации возможностей режима разработки *Windows Forms Application* на уровне обработки событий.

1. **Формулировка задачи**

Реализовать смену цвета фона экранной формы (*Form*) по нажатии кнопкой мыши (*Click*) на экранную кнопку (*Button*). Продублировать возможность смены цвета фона по выбранной опции (*RadioButton*). Опции группируются рамкой (*GroupBox*), которая в начальный момент времени после запуска программы невидима (*.Visible = false*). Смена режима управления меняется по логическому признаку (*CheckBox*): если «галочка» снята (*.Checked = false*) – управление производится экранными кнопками (элементы в составе рамки (*GroupBox*) при этом нефункциональны: *.Enabled = false*), если «галочка» выставлена (*.Checked = true*) – управление производится опциями (*RadioButton*). Предусмотреть следующий набор цветов: «Красный», «Зелёный», «Синий», «Фиолетовый», «Оранжевый», «Белый», «Стандартный» («Серый»). Организовать последний в качестве *enum*.

1. **Сеть Петри**

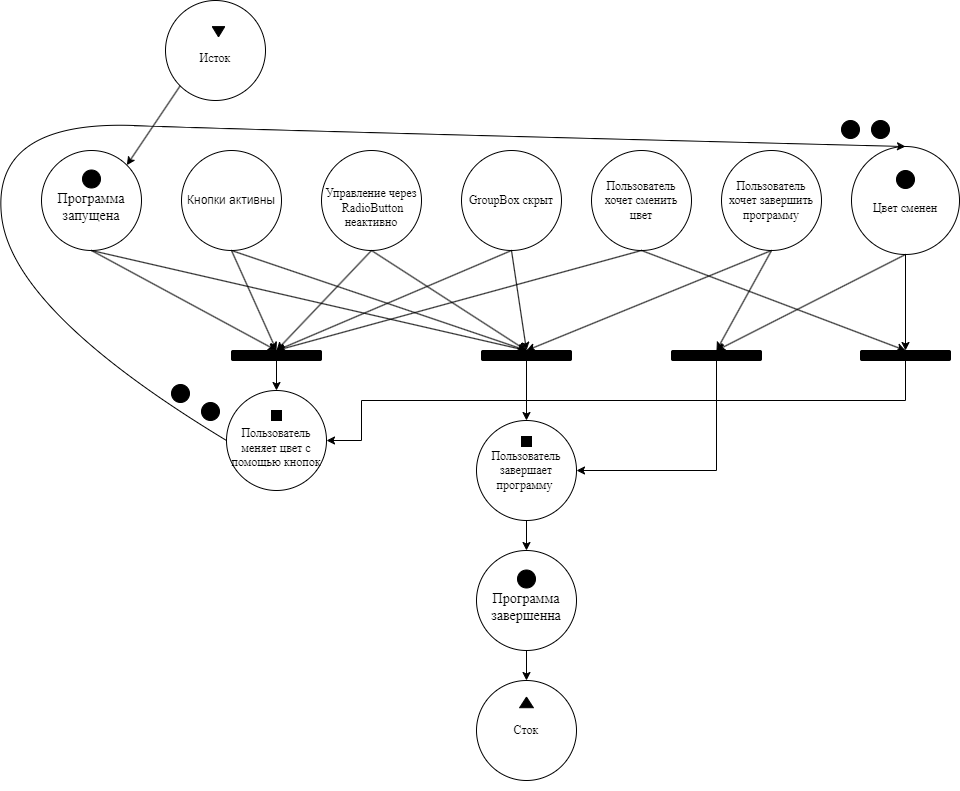
****

Рисунок 3.1.1 – Сеть Петри (смена цвета через экранную кнопку)

Раскрытие эффектов схемами алгоритма:

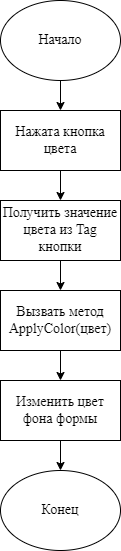


Рисунок 3.1.2 – Раскрытие эффекта: пользователь меняет цвет с помощью кнопок

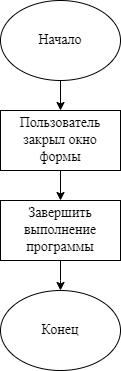


Рисунок 3.1.3 – Раскрытие эффекта: пользователь завершает программу

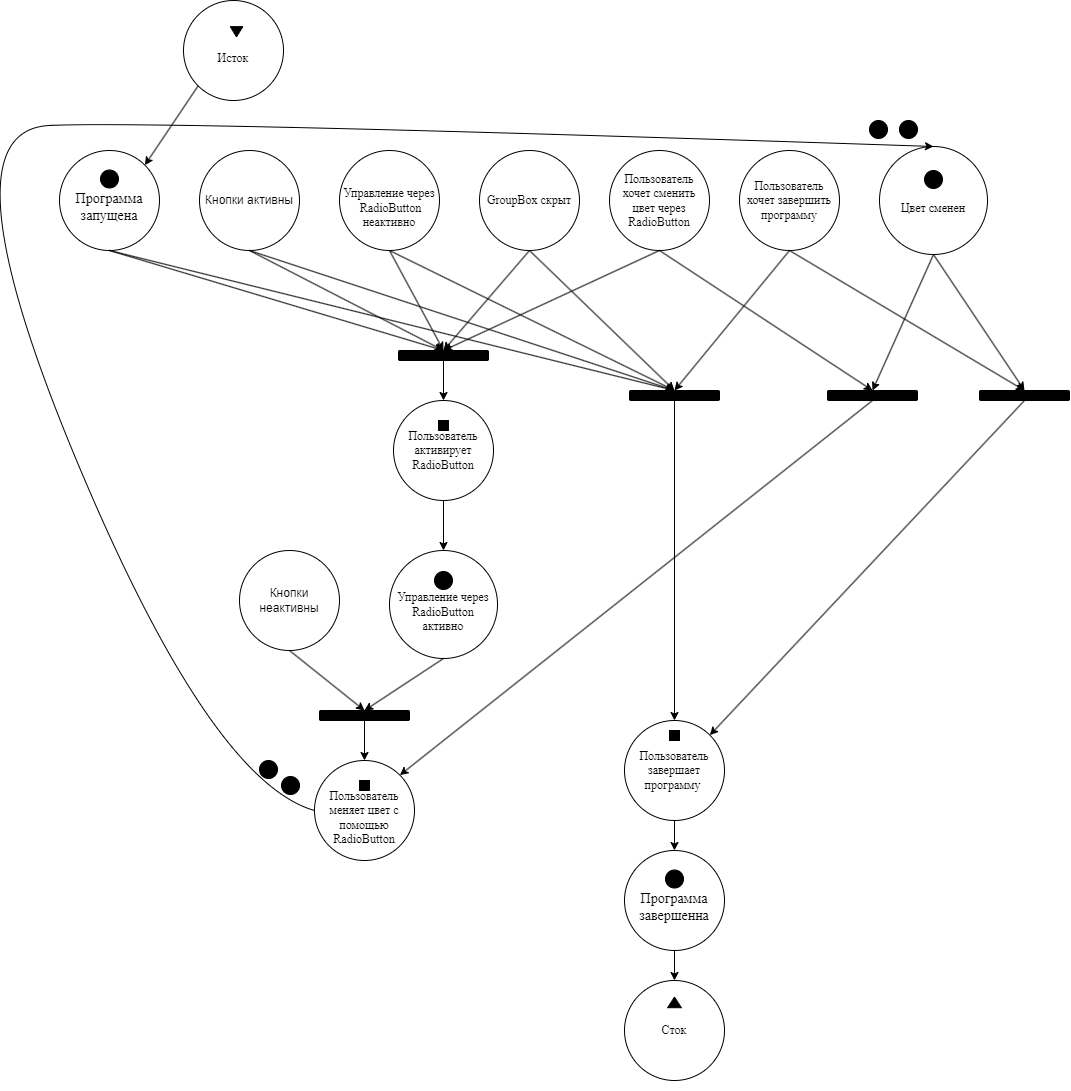


Рисунок 3.1.4 – Сеть Петри (смена цвета через *RadioButton*)

Раскрытие эффектов схемами алгоритма:

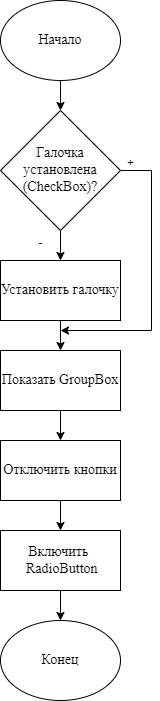


Рисунок 3.1.5 – Раскрытие эффекта: пользователь активирует *RadioButon*

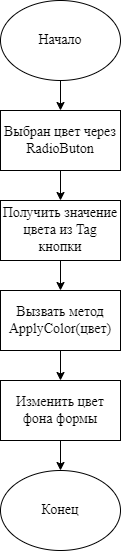


Рисунок 3.1.6 – Раскрытие эффекта: пользователь меняет цвет с помощью *RadioButon*

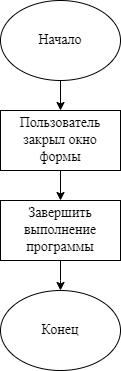


Рисунок 3.1.7 – Раскрытие эффекта: пользователь завершает программу

1. **Подбор тестовых примеров**

**Тест 1: Начальное состояние**

* Запуск приложения → Фон серый, GroupBox "Выбор цвета" скрыт
* Кнопки цветов ("Красный", "Зелёный" и т.д.) активны

**Тест 2: Смена цвета через кнопки**

* Нажатие "Красный" → фон красный
* Нажатие "Синий" → фон синий
* Нажатие "Стандартный" → фон серый

**Тест 3: Активация режима RadioButton**

* Установка флажка → GroupBox появляется
* Кнопки цветов блокируются

**Тест 4: Выбор цвета через RadioButton**

* Выбор "Фиолетовый" → фон фиолетовый
* Выбор "Оранжевый" → фон оранжевый

**Тест 5: Возврат в режим кнопок**

* Выбор "Белый" в RadioButton → фон белый
* Снятие флажка → GroupBox скрывается
* Нажатие "Зелёный" → фон зелёный

**Тест 6: Блокировка элементов**

* В режиме RadioButton: кнопки цветов неактивны
* В режиме кнопок: RadioButton недоступны

1. **Листинг (код программы)**

Program.cs:

using System;

using System.Windows.Forms;

namespace ColorChangerApp

{

static class Program

{

[STAThread]

static void Main()

{

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

ColorChangerForm сolorсhangerForm = new ColorChangerForm();

Application.Run(сolorсhangerForm);

}

}

}

ColorChangerForm.cs:

using System;

using System.Drawing;

using System.Windows.Forms;

namespace ColorChangerApp

{

public partial class ColorChangerForm : Form

{

// Перечисление для цветов

private enum AppColor

{

Красный,

Зелёный,

Синий,

Фиолетовый,

Оранжевый,

Белый,

Стандартный

}

// Элементы управления

private CheckBox modeCheckBox;

private GroupBox colorGroup;

private Button[] colorButtons;

private RadioButton[] colorRadioButtons;

public ColorChangerForm()

{

InitializeComponent();

InitializeCustomComponents();

}

private void InitializeCustomComponents()

{

// Инициализация CheckBox

modeCheckBox = new CheckBox

{

Text = "Управление через RadioButton",

Location = new Point(20, 20),

AutoSize = true

};

modeCheckBox.CheckedChanged += ModeCheckBox\_CheckedChanged;

Controls.Add(modeCheckBox);

// Получение названий цветов из перечисления

string[] colors = Enum.GetNames(typeof(AppColor));

colorButtons = new Button[colors.Length];

// Инициализация кнопок цветов

for (int i = 0; i < colors.Length; i++)

{

colorButtons[i] = new Button

{

Text = colors[i],

Location = new Point(20, 60 + i \* 40),

Size = new Size(120, 30),

Tag = Enum.Parse(typeof(AppColor), colors[i]) // Сохраняем значение enum

};

colorButtons[i].Click += ColorButton\_Click;

Controls.Add(colorButtons[i]);

}

// Инициализация GroupBox и RadioButton

colorGroup = new GroupBox

{

Text = "Выбор цвета",

Location = new Point(200, 20),

Size = new Size(150, 250),

Visible = false

};

colorRadioButtons = new RadioButton[colors.Length];

for (int i = 0; i < colors.Length; i++)

{

colorRadioButtons[i] = new RadioButton

{

Text = colors[i],

Location = new Point(10, 20 + i \* 30),

AutoSize = true,

Enabled = false,

Tag = Enum.Parse(typeof(AppColor), colors[i]) // Сохраняем значение enum

};

colorRadioButtons[i].CheckedChanged += ColorRadioButton\_CheckedChanged;

colorGroup.Controls.Add(colorRadioButtons[i]);

}

Controls.Add(colorGroup);

}

// Обработчик изменения CheckBox

private void ModeCheckBox\_CheckedChanged(object sender, EventArgs e)

{

bool isRadioMode = modeCheckBox.Checked;

colorGroup.Visible = isRadioMode;

// Включаем/выключаем кнопки в зависимости от режима

foreach (Button btn in colorButtons)

{

btn.Enabled = !isRadioMode;

}

// Включаем/выключаем RadioButton в зависимости от режима

foreach (RadioButton rb in colorRadioButtons)

{

rb.Enabled = isRadioMode;

}

}

// Обработчик клика по кнопке

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

{

Button btn = (Button)sender;

ApplyColor((AppColor)btn.Tag); // Используем сохраненное значение enum

}

// Обработчик выбора RadioButton

private void ColorRadioButton\_CheckedChanged(object sender, EventArgs e)

{

RadioButton rb = (RadioButton)sender;

if (rb.Checked)

{

ApplyColor((AppColor)rb.Tag); // Используем сохраненное значение enum

}

}

// Метод изменения цвета фона с использованием enum

private void ApplyColor(AppColor color)

{

switch (color)

{

case AppColor.Красный:

BackColor = Color.Red;

break;

case AppColor.Зелёный:

BackColor = Color.Green;

break;

case AppColor.Синий:

BackColor = Color.Blue;

break;

case AppColor.Фиолетовый:

BackColor = Color.Purple;

break;

case AppColor.Оранжевый:

BackColor = Color.Orange;

break;

case AppColor.Белый:

BackColor = Color.White;

break;

case AppColor.Стандартный:

BackColor = Color.Gray;

break;

}

}

}

}

ColorChangerForm.Designer.cs

namespace ColorChangerApp

{

partial class ColorChangerForm

{

private System.ComponentModel.IContainer components = null;

protected override void Dispose(bool disposing)

{

if (disposing && (components != null))

{

components.Dispose();

}

base.Dispose(disposing);

}

private void InitializeComponent()

{

this.SuspendLayout();

//

// ColorChangerForm

//

this.ClientSize = new System.Drawing.Size(400, 400);

this.Name = "Form1";

this.Text = "Color Changer";

this.ResumeLayout(false);

}

}

}

1. **Расчёт тестовых примеров на ПК**

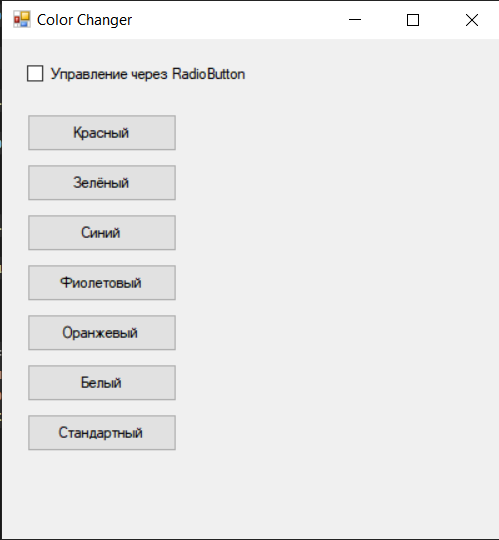
****

Рисунок 6.1.1 – Тестовый пример №1

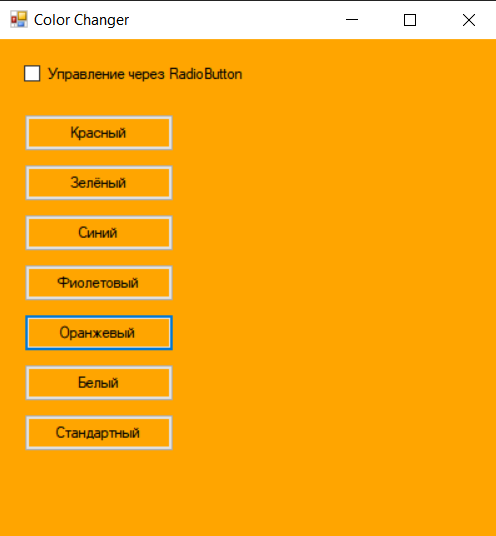
****

Рисунок 6.1.2 – Тестовый пример №2

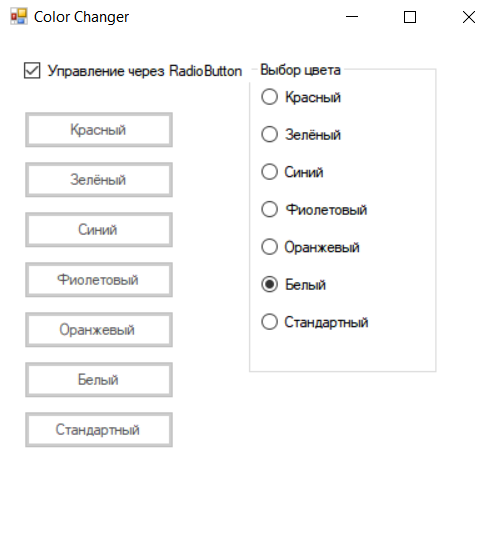
****

Рисунок 6.1.3 – Тестовый пример №3

1. **Вывод**

В ходе выполнения лабораторной работы была разработана учебная *Windows Forms*-программа с графическим пользовательским интерфейсом, демонстрирующая основы работы с событиями, перечислениями и элементами управления в среде *Windows Forms*. Приложение реализует смену цвета фона формы двумя способами: с помощью кнопок и переключателей (*RadioButton*), в зависимости от установленного режима (*CheckBox*).

Программа позволяет гибко переключаться между режимами управления, обеспечивает блокировку неактуальных элементов и корректно реагирует на пользовательский ввод. Таким образом, цель лабораторной работы достигнута, получены навыки создания пользовательских интерфейсов, обработки событий и использования перечислений.