

Министерство науки и высшего образования Российской Федерации
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Российский экономический университет имени Г.В. Плеханова»

Высшая школа кибертехнологий, математики и статистики
Кафедра информатики
Направление 38.03.05 Бизнес-информатика
Профиль «Цифровая трансформация бизнеса»

ОТЧЕТ

По выполнению практической работы №4

Выполнила:
студент группы 15.27Д-БИ19/226
3 курса ВШКМиС
Нгуен Као Бач

Москва-2024

Вариант 2

Создать меню с командами Size, Color, Paint, Quit. Команда Paint недоступна. При выборе команды Quit приложение завершается. При выборе команды Size открывается диалоговое окно, содержащее:

- два поля типа TextBox для ввода длин сторон прямоугольника;
- группу из трех флажков (Red, Green, Blue) типа CheckBox;
- кнопку типа Button. Обеспечить возможность: • ввода длин сторон прямоугольника в пикселах в поля ввода;
- выбора его цвета с помощью флажков.

После задания параметров команда Paint становится доступной. При выборе команды Paint в главном окне приложения выводится прямоугольник заданного размера и сочетания цветов или выдается сообщение, если введенные размеры превышают размер окна.

Анализ задания:

1. Создайте меню с командами: **Size, Color, Paint, Quit.**

- **Меню Size:** При выборе этой команды появится новое диалоговое окно. Это окно включает:
 - Два поля ввода (TextBox), чтобы пользователь мог ввести длину и ширину прямоугольника.
 - Опции выбора цвета, которые могут быть CheckBox для трех основных цветов (красный, зеленый, синий) или диалоговое окно выбора цвета (ColorDialog), чтобы пользователь мог выбрать цвет гибче.
 - Кнопка **ОК** для подтверждения введенных значений.
- **Меню Color:** Эта команда открывает палитру цветов для выбора цвета прямоугольника (если используется ColorDialog). Однако, если для выбора цвета используются только CheckBox, эта команда может быть не обязательной.
- **Меню Paint:**
 - Изначально эта команда заблокирована до тех пор, пока пользователь не введет корректные размеры и цвет.

- Когда команда **Paint** выбрана, приложение отобразит прямоугольник на главном окне с выбранными размерами и цветом.
- Если размеры прямоугольника слишком большие (превышают размеры окна), будет показано сообщение об ошибке вместо рисования.
- **Меню Quit:** Эта команда завершает приложение сразу.

2. Диалоговое окно для ввода размеров и цвета

- Диалоговое окно появляется, когда выбрана команда **Size**.
- Пользователь может:
 - Ввести ширину и высоту прямоугольника в поля ввода (TextBox).
 - Выбрать цвет с помощью флажков CheckBox (для трех основных цветов) или использовать палитру цветов (для большей гибкости).
- После ввода данных и нажатия **ОК** приложение сохранит информацию о размере и цвете.

3. Процесс обработки команды Paint

- Когда пользователь выбрал допустимые размеры и цвет, команда **Paint** станет доступной.
- При выборе команды **Paint** программа выполнит следующие шаги:
 - Проверит, находится ли прямоугольник в пределах размеров окна.
 - Если размеры превышены, появится сообщение об ошибке.
 - Если размеры допустимы, прямоугольник будет нарисован с выбранными размерами и цветом.

Код MainForm.h

```

1  #pragma once
2  #include "SizeForm.h"
3
4  namespace Zadach4c {
5
6      using namespace System;
7      using namespace System::ComponentModel;
8      using namespace System::Collections;
9      using namespace System::Windows::Forms;
10     using namespace System::Data;
11     using namespace System::Drawing;
12
13     /// <summary>
14     /// Summary for MainForm
15     /// </summary>
16     public ref class MainForm : public System::Windows::Forms::Form
17     {
18     public:
19         MainForm(void)
20         {
21             InitializeComponent();
22             paintToolStripMenuItem->Enabled = false;
23         }
24     private:
25         int rectWidth = 0;
26         int rectHeight = 0;
27         Color rectColor = Color::Black;
28

```

```

28
29     protected:
30         ~MainForm()
31         {
32             if (components)
33             {
34                 delete components;
35             }
36         }
37     private: System::Windows::Forms::MenuStrip^ menuStrip1;
38     private: System::Windows::Forms::ToolStripMenuItem^ sizeToolStripMenuItem;
39     private: System::Windows::Forms::ToolStripMenuItem^ colorToolStripMenuItem;
40     private: System::Windows::Forms::ToolStripMenuItem^ paintToolStripMenuItem;
41     private: System::Windows::Forms::ToolStripMenuItem^ quitToolStripMenuItem;
42
43     private:
44         System::ComponentModel::Container^ components;

```

```

45
46  ✓ #pragma region Windows Form Designer generated code
47  void InitializeComponent(void)
48  {
49      this->menuStrip1 = (gcnew System::Windows::Forms::MenuStrip());
50      this->sizeToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());
51      this->colorToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());
52      this->paintToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());
53      this->quitToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());
54      this->menuStrip1->SuspendLayout();
55      this->SuspendLayout();
56
57      // Menu items setup
58      this->menuStrip1->ImageScalingSize = System::Drawing::Size(20, 20);
59      this->menuStrip1->Items->AddRange(gcnew cli::array< System::Windows::Forms::ToolStripItem^ >(4) {
60          this->sizeToolStripMenuItem, this->colorToolStripMenuItem, this->paintToolStripMenuItem, this->quitToolStripMenuItem
61      });
62      this->menuStrip1->Location = System::Drawing::Point(0, 0);
63      this->menuStrip1->Name = L"menuStrip1";
64      this->menuStrip1->Size = System::Drawing::Size(832, 28);
65      this->menuStrip1->TabIndex = 0;
66      this->menuStrip1->Text = L"menuStrip1";
67
68      // sizeToolStripMenuItem
69      this->sizeToolStripMenuItem->Name = L"sizeToolStripMenuItem";
70      this->sizeToolStripMenuItem->Size = System::Drawing::Size(50, 24);
71      this->sizeToolStripMenuItem->Text = L"Size";
72      this->sizeToolStripMenuItem->Click += gcnew System::EventHandler(this, &MainForm::sizeToolStripMenuItem_Click);
73
74      // colorToolStripMenuItem
75      this->colorToolStripMenuItem->Name = L"colorToolStripMenuItem";
76      this->colorToolStripMenuItem->Size = System::Drawing::Size(59, 24);
77      this->colorToolStripMenuItem->Text = L"Color";
78      this->colorToolStripMenuItem->Click += gcnew System::EventHandler(this, &MainForm::colorToolStripMenuItem_Click);
79
80      // paintToolStripMenuItem
81      this->paintToolStripMenuItem->Enabled = false;
82      this->paintToolStripMenuItem->Name = L"paintToolStripMenuItem";
83      this->paintToolStripMenuItem->Size = System::Drawing::Size(55, 24);
84      this->paintToolStripMenuItem->Text = L"Paint";
85      this->paintToolStripMenuItem->Click += gcnew System::EventHandler(this, &MainForm::paintToolStripMenuItem_Click);
86
87      // quitToolStripMenuItem
88      this->quitToolStripMenuItem->Name = L"quitToolStripMenuItem";
89      this->quitToolStripMenuItem->Size = System::Drawing::Size(51, 24);
90      this->quitToolStripMenuItem->Text = L"Quit";
91      this->quitToolStripMenuItem->Click += gcnew System::EventHandler(this, &MainForm::quitToolStripMenuItem_Click);
92
93      // MainForm
94      this->AutoScaleDimensions = System::Drawing::SizeF(8, 16);
95      this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;
96      this->ClientSize = System::Drawing::Size(832, 512);
97      this->Controls->Add(this->menuStrip1);
98      this->MainMenuStrip = this->menuStrip1;
99      this->Name = L"MainForm";
100     this->Text = L"MainForm";
101     this->Load += gcnew System::EventHandler(this, &MainForm::MainForm_Load);
102     this->menuStrip1->ResumeLayout(false);
103     this->menuStrip1->PerformLayout();
104     this->ResumeLayout(false);
105     this->PerformLayout();
106 }
107 #pragma endregion
108 private: System::Void MainForm_Load(System::Object^ sender, System::EventArgs^ e) {}
109
110 private: System::Void sizeToolStripMenuItem_Click(System::Object^ sender, System::EventArgs^ e) {}

```

```

110 private: System::Void sizeToolStripMenuItem_Click(System::Object^ sender, System::EventArgs^ e) {
111     SizeForm^ sizeForm = gcnew SizeForm();
112     if (sizeForm->ShowDialog() == System::Windows::Forms::DialogResult::OK) {
113         try {
114             rectWidth = sizeForm->GetWidth();
115             rectHeight = sizeForm->GetHeight();
116             rectColor = sizeForm->GetSelectedColor();
117
118             if (rectWidth <= 0 || rectHeight <= 0) {
119                 MessageBox::Show("Пожалуйста, введите правильный размер.", "Ошибка!!", MessageBoxButtons::OK, MessageBoxIcon::Error);
120                 paintToolStripMenuItem->Enabled = false;
121             }
122             else {
123                 paintToolStripMenuItem->Enabled = true;
124             }
125         }
126         catch (...) {
127             MessageBox::Show("Пожалуйста, введите правильный размер.", "Ошибка!!", MessageBoxButtons::OK, MessageBoxIcon::Error);
128         }
129     }
130 }
131
132 private: System::Void paintToolStripMenuItem_Click(System::Object^ sender, System::EventArgs^ e) {
133     Graphics^ g = this->CreateGraphics();
134     SolidBrush^ brush = gcnew SolidBrush(rectColor);
135
136     if (rectWidth > this->ClientSize.Width || rectHeight > this->ClientSize.Height) {
137         MessageBox::Show("Размер превышает размер окна!", "Ошибка!!", MessageBoxButtons::OK, MessageBoxIcon::Error);
138     }
139     else {
140         g->FillRectangle(brush, 50, 50, rectWidth, rectHeight);
141     }
142 }

```

```

142 }
143
144 private: System::Void quitToolStripMenuItem_Click(System::Object^ sender, System::EventArgs^ e) {
145     Application::Exit();
146 }
147
148 private: System::Void colorToolStripMenuItem_Click(System::Object^ sender, System::EventArgs^ e) {
149     ColorDialog^ colorDialog = gcnew ColorDialog();
150     if (colorDialog->ShowDialog() == System::Windows::Forms::DialogResult::OK) {
151         rectColor = colorDialog->Color;
152     }
153 }
154 };
155
156 int main(array<System::String^>^ args) {
157     Application::EnableVisualStyles();
158     Application::SetCompatibleTextRenderingDefault(false);
159     Zadach4c::MainForm mainForm;
160     Application::Run(% mainForm);
161     return 0;
162 }
163 }

```

Код SizeForm.h

```

1  #pragma once
2  namespace Zadach4c {
3      using namespace System;
4      using namespace System::ComponentModel;
5      using namespace System::Collections;
6      using namespace System::Windows::Forms;
7      using namespace System::Data;
8      using namespace System::Drawing;
9      public ref class SizeForm : public System::Windows::Forms::Form
10     {
11     public:
12         SizeForm(void)
13         {
14             InitializeComponent();
15         }
16
17     protected:
18
19         ~SizeForm()
20         {
21             if (components)
22             {
23                 delete components;
24             }
25         }
26
27     private: System::Windows::Forms::Label^ label1;
28     protected:
29     private: System::Windows::Forms::Label^ label2;
30     private: System::Windows::Forms::TextBox^ textBox1;
31     private: System::Windows::Forms::TextBox^ textBox2;
32     private: System::Windows::Forms::Label^ label3;
33     private: System::Windows::Forms::CheckBox^ checkBox1;
34     private: System::Windows::Forms::CheckBox^ checkBox2;
35     private: System::Windows::Forms::CheckBox^ checkBox3;

```

```

34     private: System::Windows::Forms::CheckBox^ checkBox3;
35     private: System::Windows::Forms::Button^ button1;
36
37     private:
38         System::ComponentModel::Container^ components;
39
40     #pragma region Windows Form Designer generated code
41
42     void InitializeComponent(void)
43     {
44         this->label1 = (gcnew System::Windows::Forms::Label());
45         this->label2 = (gcnew System::Windows::Forms::Label());
46         this->textBox1 = (gcnew System::Windows::Forms::TextBox());
47         this->textBox2 = (gcnew System::Windows::Forms::TextBox());
48         this->label3 = (gcnew System::Windows::Forms::Label());
49         this->checkBox1 = (gcnew System::Windows::Forms::CheckBox());
50         this->checkBox2 = (gcnew System::Windows::Forms::CheckBox());
51         this->checkBox3 = (gcnew System::Windows::Forms::CheckBox());
52         this->button1 = (gcnew System::Windows::Forms::Button());
53         this->SuspendLayout();
54         //
55         // label1
56         //
57         this->label1->AutoSize = true;
58         this->label1->Location = System::Drawing::Point(202, 44);
59         this->label1->Name = L"label1";
60         this->label1->Size = System::Drawing::Size(52, 16);
61         this->label1->TabIndex = 0;
62         this->label1->Text = L"WIDTH";
63         //
64         // label2
65         //
66         this->label2->AutoSize = true;
67         this->label2->Location = System::Drawing::Point(202, 124);

```

```
67 this->label2->Location = System::Drawing::Point(202, 124);
68 this->label2->Name = L"label2";
69 this->label2->Size = System::Drawing::Size(58, 16);
70 this->label2->TabIndex = 1;
71 this->label2->Text = L"HEIGHT";
72 //
73 // textBox1
74 //
75 this->textBox1->Location = System::Drawing::Point(327, 38);
76 this->textBox1->Name = L"textBox1";
77 this->textBox1->Size = System::Drawing::Size(100, 22);
78 this->textBox1->TabIndex = 2;
79 this->textBox1->Text = L"0";
80 //
81 // textBox2
82 //
83 this->textBox2->Location = System::Drawing::Point(327, 118);
84 this->textBox2->Name = L"textBox2";
85 this->textBox2->Size = System::Drawing::Size(100, 22);
86 this->textBox2->TabIndex = 3;
87 this->textBox2->Text = L"0";
88 //
89 // label3
90 //
91 this->label3->AutoSize = true;
92 this->label3->Location = System::Drawing::Point(369, 230);
93 this->label3->Name = L"label3";
94 this->label3->Size = System::Drawing::Size(39, 16);
95 this->label3->TabIndex = 4;
96 this->label3->Text = L"Color";
97 //
98 // checkBox1
99 //
100 this->checkBox1->AutoSize = true;
```



```
100 this->checkBox1->AutoSize = true;
101 this->checkBox1->Location = System::Drawing::Point(154, 314);
102 this->checkBox1->Name = L"checkBox1";
103 this->checkBox1->Size = System::Drawing::Size(58, 20);
104 this->checkBox1->TabIndex = 5;
105 this->checkBox1->Text = L"RED";
106 this->checkBox1->UseVisualStyleBackColor = true;
107 //
108 // checkBox2
109 //
110 this->checkBox2->AutoSize = true;
111 this->checkBox2->Location = System::Drawing::Point(350, 314);
112 this->checkBox2->Name = L"checkBox2";
113 this->checkBox2->Size = System::Drawing::Size(77, 20);
114 this->checkBox2->TabIndex = 6;
115 this->checkBox2->Text = L"GREEN";
116 this->checkBox2->UseVisualStyleBackColor = true;
117 //
118 // checkBox3
119 //
120 this->checkBox3->AutoSize = true;
121 this->checkBox3->Location = System::Drawing::Point(544, 314);
122 this->checkBox3->Name = L"checkBox3";
123 this->checkBox3->Size = System::Drawing::Size(64, 20);
124 this->checkBox3->TabIndex = 7;
125 this->checkBox3->Text = L"BLUE";
126 this->checkBox3->UseVisualStyleBackColor = true;
127 //
128 // button1
129 //
130 this->button1->Location = System::Drawing::Point(352, 408);
131 this->button1->Name = L"button1";
132 this->button1->Size = System::Drawing::Size(75, 23);
133 this->button1->TabIndex = 8;
```

```

133         this->button1->TabIndex = 8;
134         this->button1->Text = L"OK";
135         this->button1->UseVisualStyleBackColor = true;
136         this->button1->Click += gcnew System::EventHandler(this, &SizeForm::button1_Click);
137         //
138         // SizeForm
139         //
140         this->AutoScaleDimensions = System::Drawing::SizeF(8, 16);
141         this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;
142         this->ClientSize = System::Drawing::Size(793, 517);
143         this->Controls->Add(this->button1);
144         this->Controls->Add(this->checkBox3);
145         this->Controls->Add(this->checkBox2);
146         this->Controls->Add(this->checkBox1);
147         this->Controls->Add(this->label3);
148         this->Controls->Add(this->textBox2);
149         this->Controls->Add(this->textBox1);
150         this->Controls->Add(this->label2);
151         this->Controls->Add(this->label1);
152         this->Name = L"SizeForm";
153         this->Text = L"SizeForm";
154         this->ResumeLayout(false);
155         this->PerformLayout();
156     }
157 }
158 #pragma endregion
159 public:
160     int GetWidth() { return int::Parse(textBox1->Text); }
161     int GetHeight() { return int::Parse(textBox2->Text); }
162     Color GetSelectedColor() {
163         if (checkBox1->Checked) return Color::Red;
164         if (checkBox2->Checked) return Color::Green;
165         if (checkBox3->Checked) return Color::Blue;
166         return Color::Black; //Цвет по умолчанию, если ни один из них не выбран
167     }
168 private: System::Void button1_Click(System::Object^ sender, System::EventArgs^ e) {
169     try {
170         int width = GetWidth();
171         int height = GetHeight();
172         if (width <= 0 || height <= 0) {
173             MessageBox::Show("Пожалуйста, введите правильные целые положительные числа для ширины и высоты.", "Invalid Input", MessageBoxButtons::OK, MessageBoxIcon::Error);
174             return;
175         }
176         this->DialogResult = System::Windows::Forms::DialogResult::OK;
177         this->Close();
178     }
179     catch (Exception^ ex) {
180         MessageBox::Show("Введите правильные числовые значения для ширины и высоты.", "Invalid Input", MessageBoxButtons::OK, MessageBoxIcon::Error);
181     }
182 }
183 };
184
185

```

Файл MyForm.h

Этот код представляет графический интерфейс (GUI), реализованный на C++/CLI с использованием Windows Forms. Он определяет класс MainForm, который предоставляет основное окно приложения для рисования прямоугольника с заданными пользователем размерами и цветом.

В начале используется `#pragma once` для предотвращения многократного включения файла, а также `#include "SizeForm.h"` для подключения формы SizeForm, в которой пользователь выбирает размеры и цвет прямоугольника. Класс MainForm, наследующий `System::Windows::Forms::Form`, содержит

переменные `rectWidth`, `rectHeight`, и `rectColor`, которые хранят размеры и цвет прямоугольника. Конструктор `MainForm()` вызывает метод `InitializeComponent()`, чтобы инициализировать элементы интерфейса и отключить меню `Paint`, так как размеры прямоугольника еще не заданы.

Метод `InitializeComponent` создает и настраивает элементы интерфейса: главное меню `menuStrip1` с пунктами `Size`, `Color`, `Paint`, и `Quit`. Пункт меню `Size` открывает форму `SizeForm` для ввода размеров, `Color` позволяет выбрать цвет, `Paint` рисует прямоугольник (он отключен по умолчанию), а `Quit` завершает приложение.

Событие `sizeToolStripMenuItem_Click` вызывает `SizeForm`, где пользователь может установить размеры и цвет прямоугольника. Если размеры валидны, то `Paint` активируется, иначе показывается сообщение об ошибке, и `Paint` остается неактивным. При нажатии на `paintToolStripMenuItem_Click` проверяется, входят ли размеры в пределы окна, и, если они допустимы, прямоугольник рисуется с заданным цветом. Пункт меню `Quit` вызывает событие `quitToolStripMenuItem_Click` для закрытия приложения, а `colorToolStripMenuItem_Click` открывает `ColorDialog` для выбора цвета. Основная функция `main` запускает приложение, создавая объект `MainForm` и отображая главное окно.

Файл `SizeForm.h`

Этот код на C++/CLI с использованием `Windows Forms` определяет класс `SizeForm`, который предоставляет графический интерфейс для ввода пользователем ширины, высоты и выбора цвета прямоугольника. `SizeForm` содержит несколько элементов управления, таких как метки (`Label`), текстовые поля (`TextBox`) и флажки (`CheckBox`), чтобы пользователь мог ввести размеры и выбрать один из доступных цветов (красный, зеленый или синий). Также имеется кнопка `ОК`, по нажатию на которую проверяется корректность введенных данных.

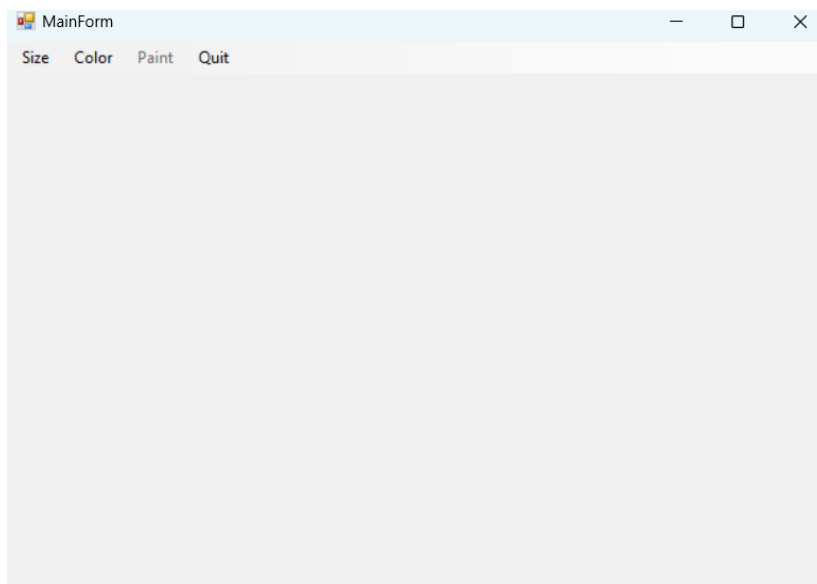
Класс `SizeForm` начинается с конструктора, который вызывает метод `InitializeComponent()` для инициализации всех компонентов интерфейса. Затем следует деструктор, который освобождает ресурсы, если они были выделены. В `InitializeComponent` создаются и настраиваются все элементы интерфейса. Метки `label1` и `label2` используются для обозначения полей ширины и высоты, текстовые поля `textBox1` и `textBox2` позволяют пользователю вводить значения

ширины и высоты, и флажки `checkBox1`, `checkBox2`, и `checkBox3` позволяют выбрать цвет (красный, зеленый или синий).

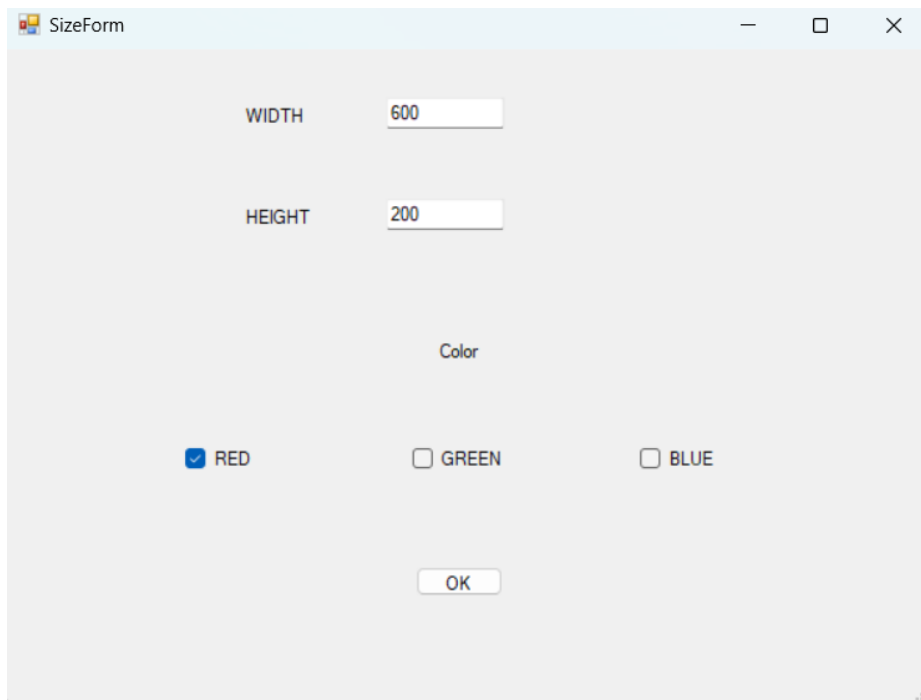
Метод `GetWidth()` возвращает значение ширины из `textBox1`, а метод `GetHeight()` — значение высоты из `textBox2`, преобразуя текст в целочисленный формат. Метод `GetSelectedColor()` проверяет, какой флажок выбран, и возвращает соответствующий цвет. Если ни один из флажков не выбран, по умолчанию возвращается черный цвет.

Обработчик события `button1_Click` срабатывает при нажатии кнопки ОК. Он вызывает методы `GetWidth()` и `GetHeight()`, чтобы получить введенные значения, и проверяет, что они больше нуля. Если введенные данные некорректны, отображается сообщение об ошибке. Если все данные верны, форма закрывается и возвращает результат `DialogResult::OK`, позволяя основному окну (например, `MainForm`) получить введенные значения.

Окно по умолчанию



Окно `SizeForm` позволяет пользователю ввести длину, ширину и цвет прямоугольника:

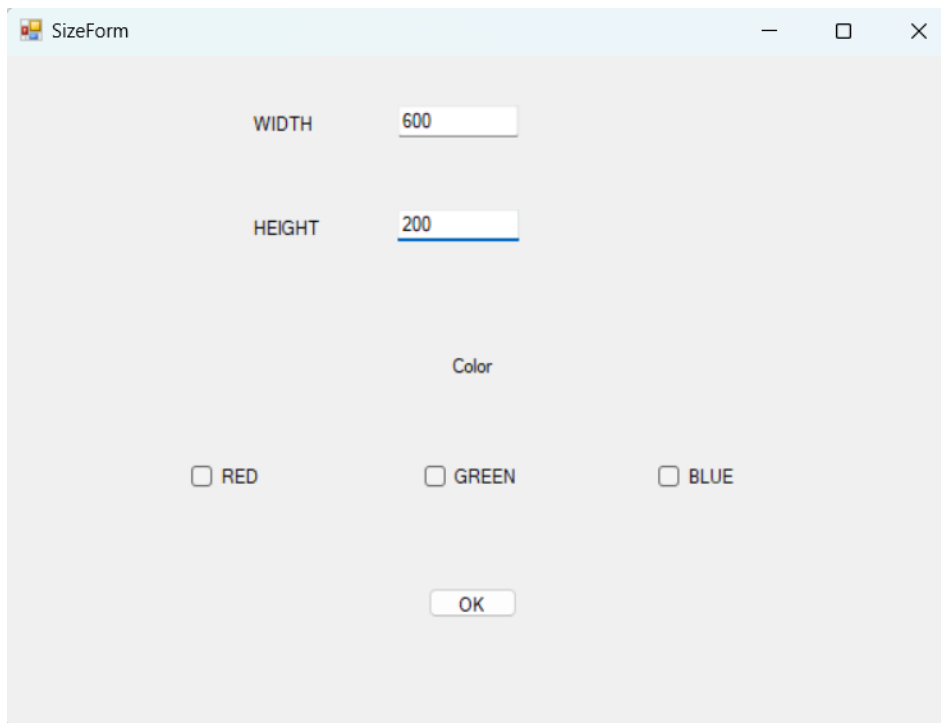


A Windows-style dialog box titled "SizeForm". It contains two input fields: "WIDTH" with the value "600" and "HEIGHT" with the value "200". Below these is a "Color" label. Under "Color", there are three checkboxes: "RED" (checked), "GREEN" (unchecked), and "BLUE" (unchecked). At the bottom is an "OK" button.

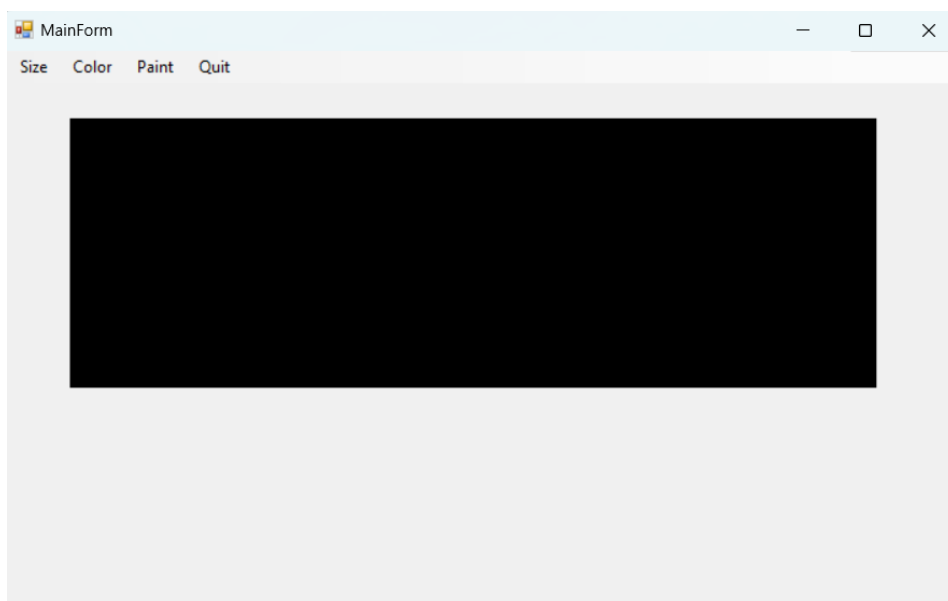
Результат при выборе длины 600 пикселей, ширина 200 пикселей, цвет КРАСНЫЙ:



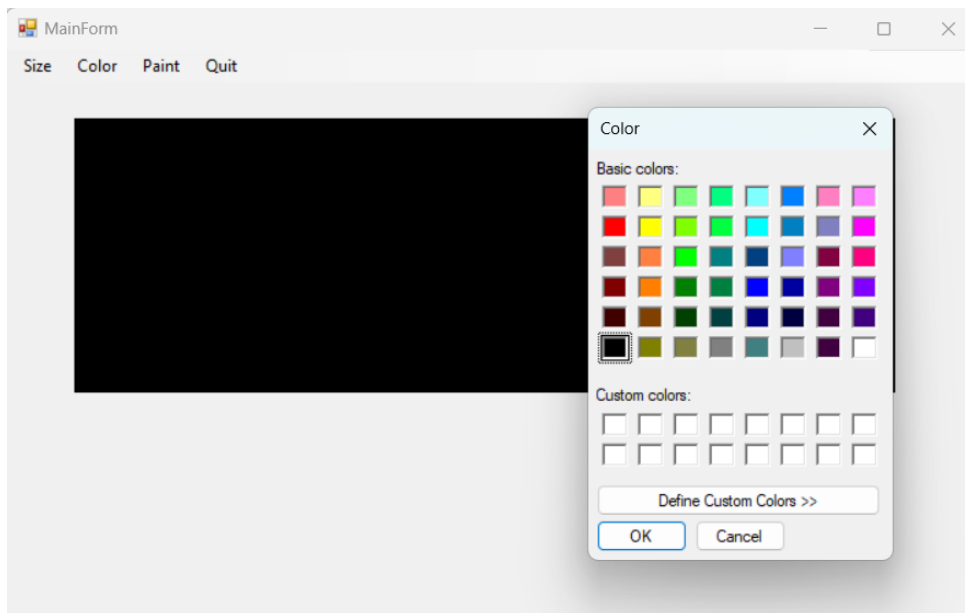
Если пользователь не выбирает цвет, программа по умолчанию печатает ЧЕРНЫМ цветом:



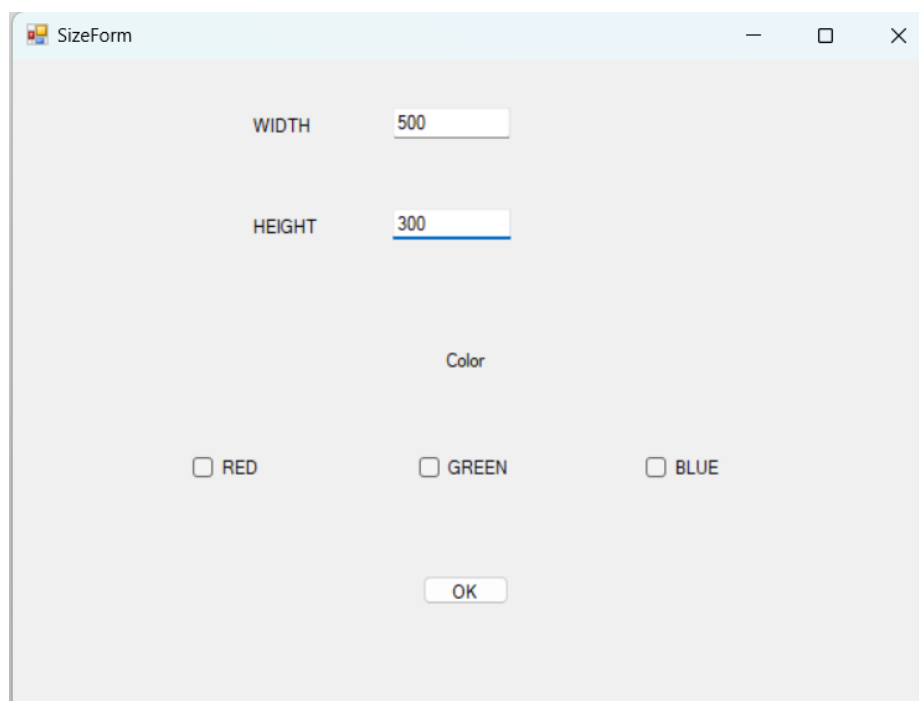
A screenshot of a Windows-style dialog box titled "SizeForm". It features a light gray background and a title bar with standard minimize, maximize, and close buttons. The dialog contains two input fields: "WIDTH" with the value "600" and "HEIGHT" with the value "200". Below these is a "Color" label, followed by three radio button options: "RED", "GREEN", and "BLUE". At the bottom center is an "OK" button.

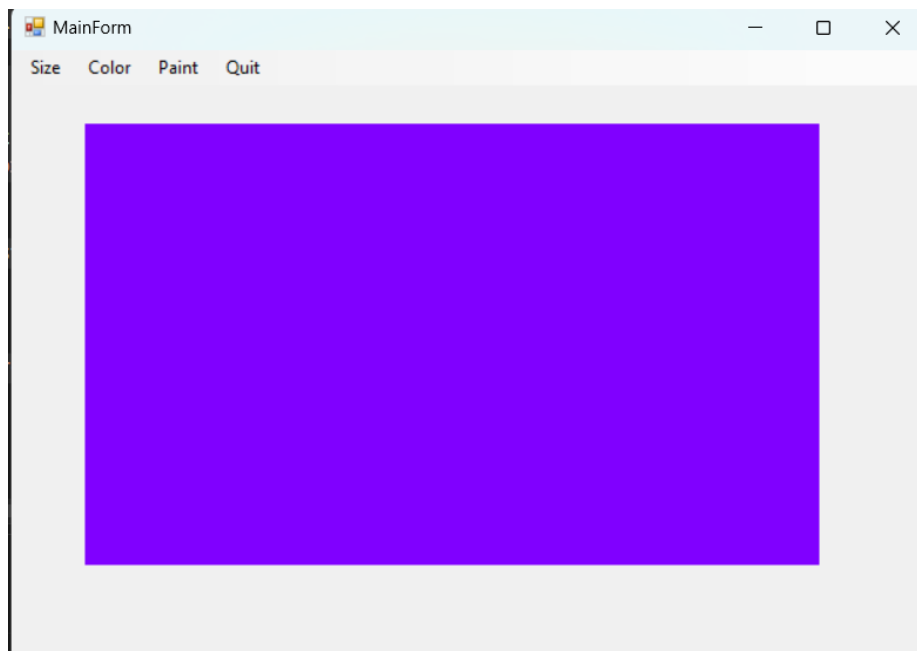


Диалоговое окно «Цвет» позволяет пользователям выбирать любой цвет:

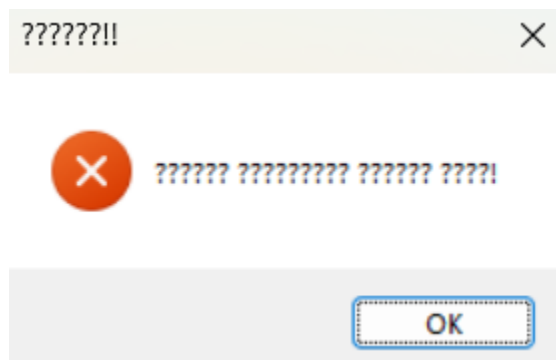


Пользователь использует ФИОЛЕТОВЫЙ цвет, ширину 300 пикселей, длину 500 пикселей:





Когда пользователь вводит ширину 500 пикселей, длину 300 пикселей. Программа отобразит сообщение, поскольку размер прямоугольника больше окна:



Если пользователь не введет длину и ширину, появится сообщение:

