

**Тема 4. Работа с элементами  
управления. Списки, комбинированные  
поля, чекбоксы и радиокнопки.  
Контейнеры и управление  
расположением компонентов.**

**Цель занятия:**

**Изучить работу с элементами управления, такими как списки, комбинированные поля, чекбоксы и радиокнопки, основными контейнерами и механизмами управления расположением компонентов.**

# **Учебные вопросы:**

- 1. Чекбоксы (CheckBox).**
- 2. Радиокнопки (RadioButton).**
- 3. Списки (ListBox).**
- 4. Комбинированные поля (ComboBox).**
- 5. Элементы управления для работы с датой и временем.**
- 6. Основные контейнеры в Windows Forms.**
- 7. Управление расположением компонентов.**
- 8. Примеры использования контейнеров.**

# 1. Чекбоксы (CheckBox).

Назначение и использование:

- Выбор одного или нескольких вариантов.
- Использование для включения/выключения опций.

Основные свойства:

- Text: текст рядом с чекбоксом.
- Checked: состояние чекбокса (выбран/не выбран).
- ThreeState: поддержка третьего состояния (неопределенное).

Основные методы:

- `Check()`: установить состояние "выбран".
- `Uncheck()`: установить состояние "не выбран".

Основные события:

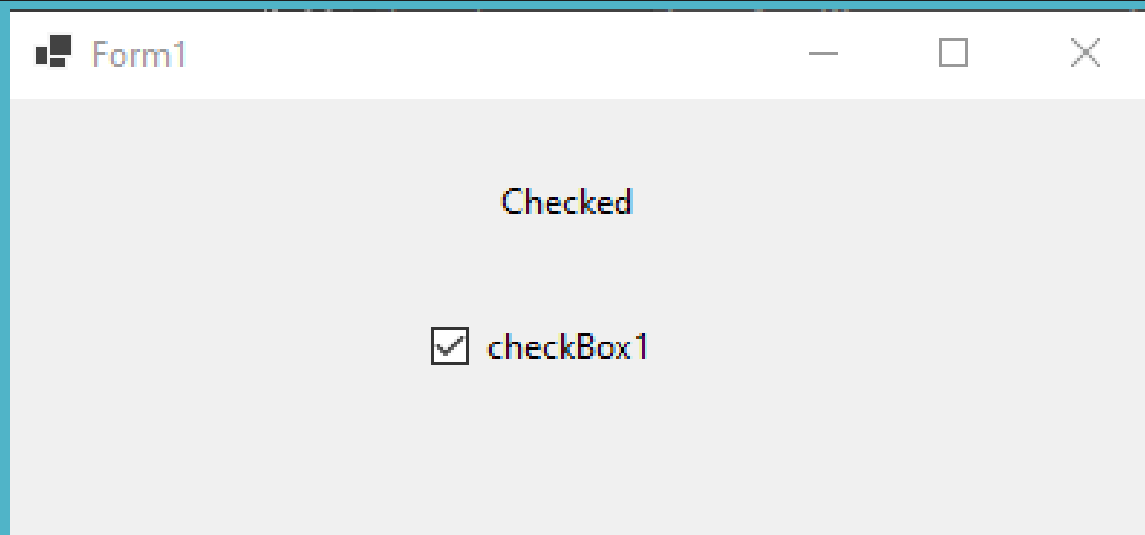
- `CheckedChanged`: изменение состояния чекбокса.

Пример использования:

- Создание группы чекбоксов.
- Обработка изменения состояния.

Пример обработки события `CheckedChanged` (изменение состояния чекбокса).

```
private void checkBox1_CheckedChanged(object sender, EventArgs e)
{
    if (checkBox1.Checked)
        label1.Text = "Checked";
    else
        label1.Text = "Unchecked";
}
```



The screenshot shows a standard Windows application window titled "Form1". Inside the window, there is a label with the text "Checked" and a checkbox labeled "checkBox1". The checkbox is currently checked, with a small square containing a checkmark to its left.

## 2. Радиокнопки (RadioButton).

Назначение и использование:

- Выбор одного варианта из группы.
- Использование для взаимоисключающих опций.

Основные свойства:

- Text: текст рядом с радиокнопкой.
- Checked: состояние радиокнопки (выбрана/не выбрана).
- AutoCheck: автоматическое снятие выбора с других радиокнопок в группе.

Основные методы:

- `Check()`: установить состояние "выбрана".
- `Uncheck()`: установить состояние "не выбрана".

Основные события:

- `CheckedChanged`: изменение состояния радиокнопки.

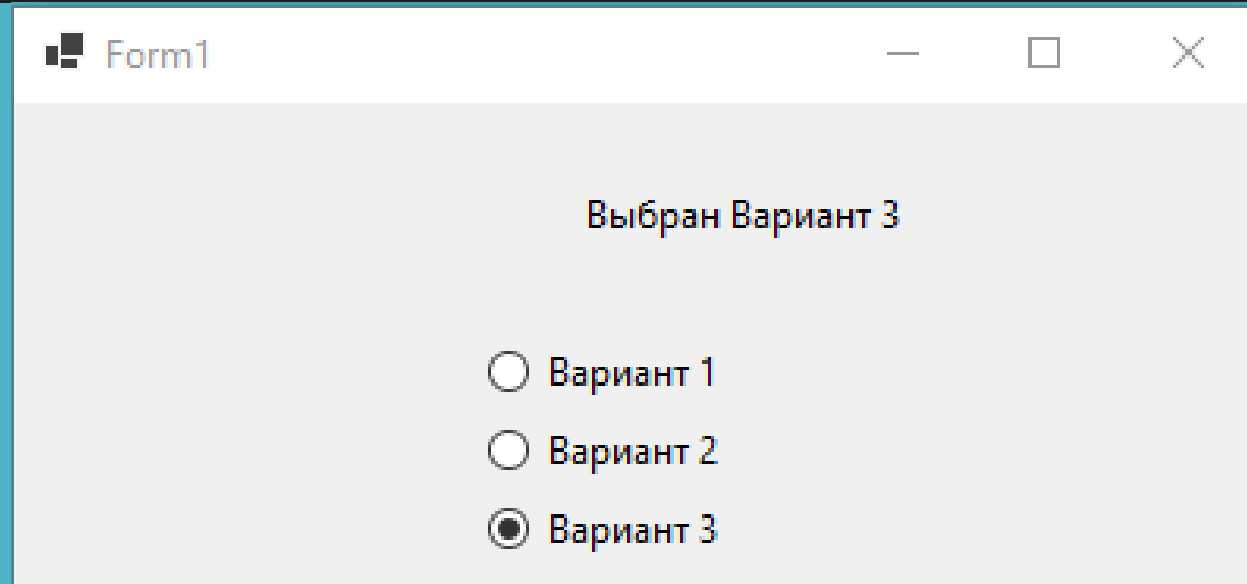
Пример использования:

- Создание группы радиокнопок.
- Обработка выбора радиокнопки.



# Пример обработки события CheckedChanged (изменение состояния радиокнопки).

```
private void radioButton_CheckedChanged(object sender, EventArgs e)
{
    // Проверяем, какая радиокнопка выбрана
    if (radioButton1.Checked)
        label1.Text = "Выбран Вариант 1";
    else if (radioButton2.Checked)
        label1.Text = "Выбран Вариант 2";
    else
        label1.Text = "Выбран Вариант 3";
}
```



The screenshot shows a Windows application window titled "Form1". Inside the window, there is a label displaying the text "Выбран Вариант 3". Below the label, there are three radio buttons arranged vertically, each followed by its respective label: "Вариант 1", "Вариант 2", and "Вариант 3". The radio button for "Вариант 3" is selected, indicated by a filled circle.

# 3. Списки (ListBox).

**Назначение и использование:**

- Отображение списка элементов.
- Поддержка множественного выбора.

**Основные свойства:**

- Items: коллекция элементов.
- SelectedItem: выбранный элемент.
- SelectedIndex: индекс выбранного элемента.
- SelectionMode: режим выбора (один или несколько элементов).
- MultiColumn: отображение в несколько колонок.

## **Основные методы:**

- `Items.Add()`: добавление элемента.
- `Items.Remove()`: удаление элемента.
- `Items.Clear()`: очистка списка.

## **Основные события:**

- `SelectedIndexChanged`: изменение выбранного элемента.
- `SelectedValueChanged`: изменение значения выбранного элемента.

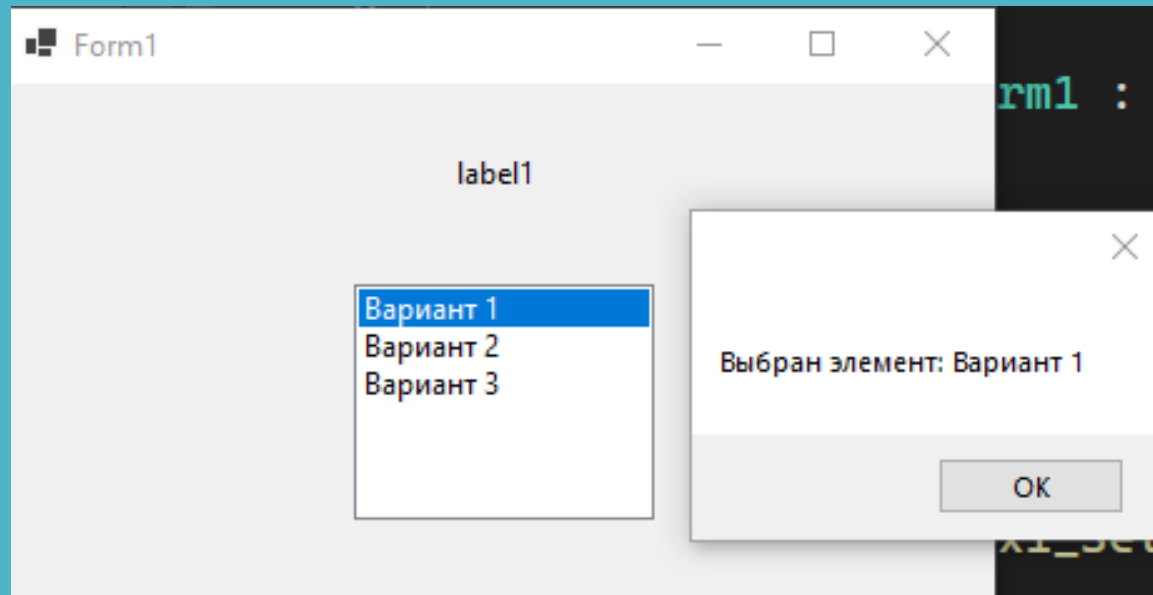
## **Пример использования:**

- Добавление элементов в список.
- Обработка выбора элемента.

# Пример обработки события SelectedIndexChanged (изменение выбранного элемента).

```
Ссылка: 1
private void listBox1_SelectedIndexChanged(object sender, EventArgs e)
{
    // Проверяем, выбран ли элемент
    if (listBox1.SelectedItem != null)
    {
        // Получаем выбранный элемент
        string selectedItem = listBox1.SelectedItem.ToString();

        // Выводим сообщение с выбранным элементом
        MessageBox.Show("Выбран элемент: " + selectedItem);
    }
}
```



## 4. Комбинированные поля (ComboBox).

**Назначение и использование:**

- Выпадающий список для выбора одного элемента.

**Основные свойства:**

- Items: коллекция элементов.
- SelectedItem: выбранный элемент.
- SelectedIndex: индекс выбранного элемента.
- DropDownStyle: стиль списка (простой, выпадающий, выпадающий с редактированием).
- Text: текст в поле ввода.

## **Основные методы:**

- `Items.Add()`: добавление элемента.
- `Items.Remove()`: удаление элемента.
- `Items.Clear()`: очистка списка.

## **Основные события:**

- `SelectedIndexChanged`: изменение выбранного элемента.
- `TextChanged`: изменение текста в поле ввода (если `ComboBox` редактируемый).

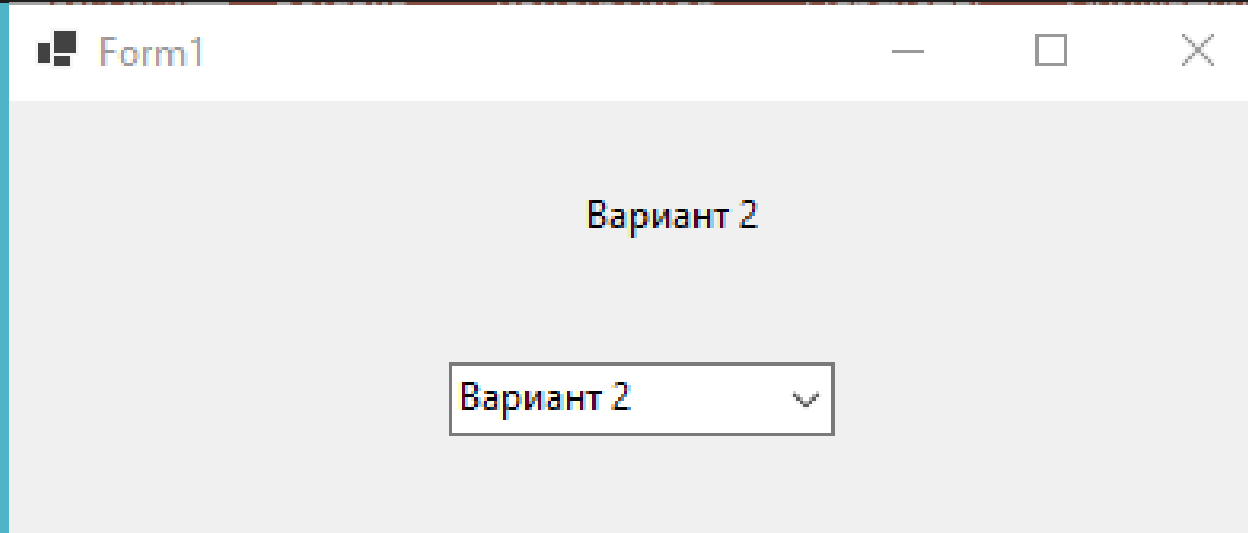
## **Пример использования:**

- Добавление элементов в `ComboBox`.
- Обработка выбора элемента.

# Пример обработки события SelectedIndexChanged (изменение выбранного элемента).

```
private void comboBox1_SelectedIndexChanged(object sender, EventArgs e)
{
    // Проверяем, выбран ли элемент
    if (comboBox1.SelectedItem != null)
    {
        // Получаем выбранный элемент
        string selectedItem = comboBox1.SelectedItem.ToString();

        // Выводим сообщение с выбранным элементом
        label1.Text = selectedItem;
    }
}
```



The screenshot shows a Windows application window titled "Form1". Inside the window, there is a label with the text "Вариант 2". Below the label is a dropdown menu (ComboBox) that also displays "Вариант 2" and has a downward arrow on its right side, indicating it is the selected item.

# 5. Элементы управления для работы с датой и временем.

## **DateTimePicker**

### **Назначение:**

DateTimePicker позволяет пользователю выбирать дату и/или время из выпадающего календаря или вводить их вручную.

Поддерживает различные форматы отображения даты и времени.



## Основные свойства:

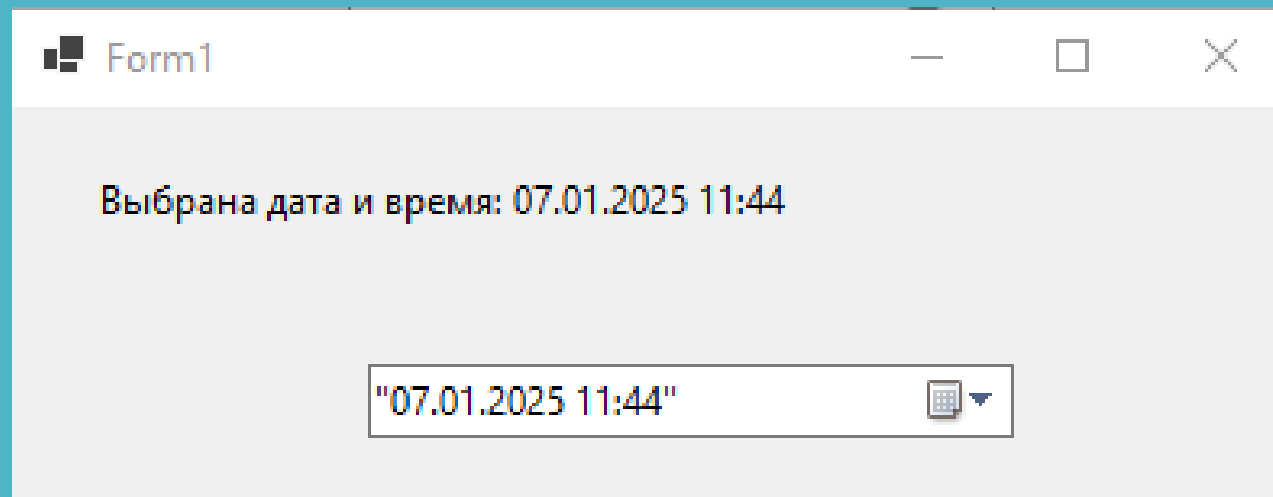
- Value: текущая выбранная дата и время.
- Format: формат отображения даты и времени (например, Short, Long, Time, Custom).
- CustomFormat: пользовательский формат (например, dd.MM.yyyy HH:mm).
- ShowUpDown: отображение стрелок для изменения значений (вместо выпадающего календаря).
- MinDate и MaxDate: минимальная и максимальная допустимая дата.

## Основные события:

- ValueChanged: происходит при изменении выбранной даты или времени.
- CloseUp: происходит при закрытии выпадающего календаря.

# Пример обработки события DateSelected (выбор даты/времени)

```
Семейство 1
private void dateTimePicker1_ValueChanged(object sender, EventArgs e)
{
    DateTime selectedDateTime = dateTimePicker1.Value;
    label1.Text = "Выбрана дата и время: " + selectedDateTime.ToString("dd.MM.yyyy HH:mm");
}
```



The screenshot shows a Windows Form titled "Form1". Inside the form, a label displays the text "Выбрана дата и время: 07.01.2025 11:44". Below the label, there is a date and time picker control. The text box of the picker shows "07.01.2025 11:44", and to its right is a calendar icon and a dropdown arrow.

# MonthCalendar

## Назначение:

- MonthCalendar позволяет пользователю выбирать дату или диапазон дат из календаря.
- Подходит для выбора одной даты или диапазона дат (например, для бронирования).

## **Основные свойства:**

- **SelectionStart**: начальная дата выбранного диапазона.
- **SelectionEnd**: конечная дата выбранного диапазона.
- **SelectionRange**: выбранный диапазон дат.
- **MaxSelectionCount**: максимальное количество дней, которые можно выбрать.
- **MinDate** и **MaxDate**: минимальная и максимальная допустимая дата.

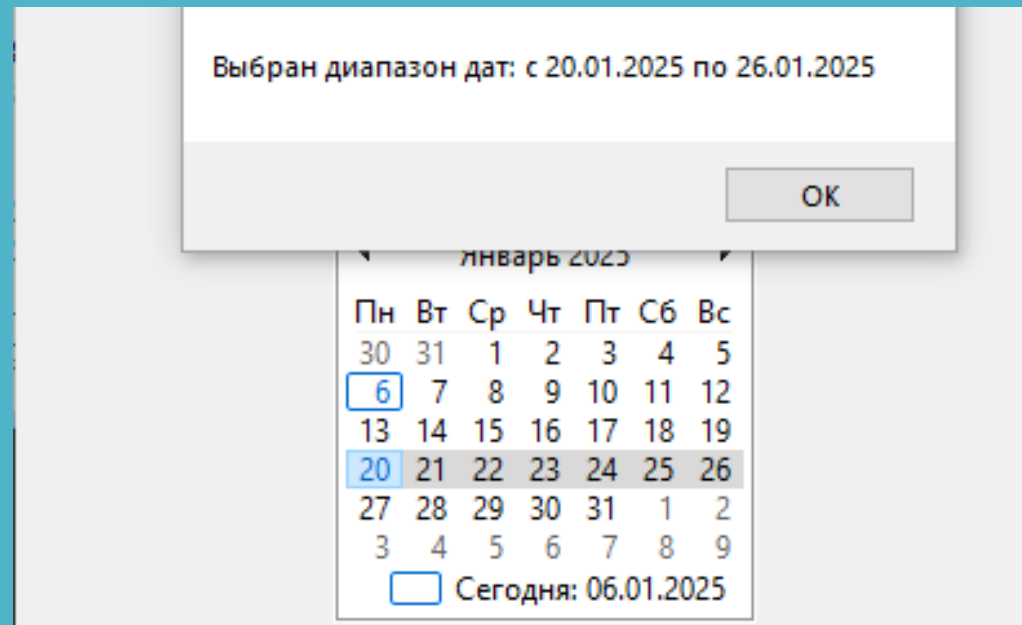
## **Основные события:**

- **DateChanged**: происходит при изменении выбранной даты.
- **DateSelected**: происходит при выборе даты пользователем.

# Пример обработки события DateSelected (выбор диапазона дат)

```
Ссылка: 1
private void monthCalendar1_DateSelected(object sender, DateRangeEventArgs e)
{
    DateTime startDate = monthCalendar1.SelectionStart;
    DateTime endDate = monthCalendar1.SelectionEnd;

    if (startDate == endDate)
        MessageBox.Show("Выбрана дата: " + startDate.ToString("dd.MM.yyyy"));
    else
        MessageBox.Show("Выбран диапазон дат: с " + startDate.ToString("dd.MM.yyyy")
            + " по " + endDate.ToString("dd.MM.yyyy"));
}
```



## 6. Основные контейнеры в Windows Forms.

Контейнеры в Windows Forms — это специальные элементы управления, которые используются для группировки и организации других элементов управления (кнопок, текстовых полей, меток и т.д.).

Они выступают в роли "родительских" элементов, внутри которых размещаются "дочерние" элементы.

## Основные задачи контейнеров:

1. Группировка элементов управления
2. Управление расположением элементов
3. Создание сложных и адаптивных интерфейсов
4. Упрощение управления элементами
5. Улучшение читаемости и поддержки кода

В Windows Forms существует несколько стандартных контейнеров, которые используются для группировки и управления расположением элементов управления.

Каждый из этих контейнеров имеет свои особенности и применяется в зависимости от задач. Рассмотрим основные контейнеры:



# 1. Panel – простая панель

Описание:

Panel — это простой контейнер, который используется для группировки элементов управления. Он не имеет визуальных эффектов, таких как рамка или заголовков, но может быть настроен с помощью свойств (например, цвет фона или границы).

**Основные свойства:**

BackColor – цвет фона панели.

BorderStyle – стиль границы (None, FixedSingle, Fixed3D).

AutoScroll – автоматическая прокрутка, если содержимое не помещается в панель.

## **2. GroupBox – группа с заголовком**

Используется для логической группировки связанных элементов управления (например, радиокнопок или флажков).

### **Основные свойства:**

Text – заголовок группы.

FlatStyle – стиль отображения рамки (Standard, Flat, Popup, System).

### **3. TabControl – контейнер с вкладками**

TabControl позволяет создавать интерфейсы с вкладками. Каждая вкладка (TabPage) является отдельным контейнером, который может содержать свои элементы управления.

#### **Основные свойства:**

TabPage – коллекция вкладок.

SelectedIndex – индекс активной вкладки.

## **4. FlowLayoutPanel – автоматическое расположение в потоке**

FlowLayoutPanel автоматически располагает элементы управления в потоке (горизонтально или вертикально). Это полезно для создания адаптивных интерфейсов.

### **Основные свойства:**

FlowDirection – направление расположения элементов (LeftToRight, RightToLeft, TopDown, BottomUp).

WrapContents – перенос элементов на новую строку/столбец, если они не помещаются.

## **5. TableLayoutPanel – табличное расположение**

TableLayoutPanel позволяет размещать элементы управления в таблице с заданным количеством строк и столбцов. Это полезно для создания сложных макетов.

### **Основные свойства:**

RowCount – количество строк.

ColumnCount – количество столбцов.

RowStyles и ColumnStyles – настройки стилей строк и столбцов (размеры, автоматическое растяжение).

# Различия между TableLayoutPanel и FlowLayoutPanel

Характеристика	TableLayoutPanel	FlowLayoutPanel
Принцип размещения	Размещает элементы в виде таблицы с фиксированным числом строк и столбцов	Располагает элементы последовательно в строку или столбец
Автоматическая адаптация	Можно задавать фиксированные размеры строк/столбцов или использовать автоматическое масштабирование	Автоматически переносит элементы на новую строку/столбец, если не хватает места
Гибкость компоновки	Хорошо подходит для сложных макетов, где важно точное позиционирование элементов	Подходит для динамического размещения элементов, когда их количество заранее неизвестно
Поддержка относительных размеров	Поддерживает настройку ширины/высоты в процентах	Все элементы имеют свои собственные размеры и размещаются без относительных привязок
Применение в адаптивных интерфейсах	Используется, когда требуется точная табличная структура (например, формы, настройки)	Хорош для панелей инструментов, динамических списков и адаптивных интерфейсов

## 6. **SplitContainer** – разделение области

SplitContainer разделяет область на две части, между которыми можно изменять размеры. Это полезно для создания интерфейсов с разделенными областями (например, список файлов и область просмотра).

### **Основные свойства:**

Orientation – ориентация разделения (Horizontal или Vertical).

Panel1 и Panel2 – две разделенные области.

# 7. Управление расположением компонентов.

Управление расположением компонентов — это важный аспект разработки пользовательских интерфейсов.

В Windows Forms для этого используются различные свойства и контейнеры, которые позволяют гибко настраивать размещение элементов управления на форме.

Рассмотрим основные механизмы управления расположением.



## **1. Свойство Dock – закрепление элементов**

Свойство Dock позволяет закрепить элемент управления относительно краев контейнера (формы или другого контейнера). Элемент "прилипает" к указанному краю и растягивается вдоль него.

### **Возможные значения:**

None – элемент не закреплен (по умолчанию).

Top – закрепление к верхнему краю.

Bottom – закрепление к нижнему краю.

Left – закрепление к левому краю.

Right – закрепление к правому краю.

Fill – заполнение всего доступного пространства.

## **2. Свойство Anchor – привязка элементов**

Свойство Anchor позволяет привязать элемент управления к одному или нескольким краям контейнера. При изменении размеров контейнера элемент будет сохранять расстояние до указанных краев.

### **Возможные значения:**

Top – привязка к верхнему краю.

Bottom – привязка к нижнему краю.

Left – привязка к левому краю.

Right – привязка к правому краю.

### **3. Свойство AutoSize – автоматическое изменение размеров**

Свойство AutoSize позволяет элементу управления автоматически изменять свои размеры в зависимости от содержимого.

### **4. Свойства Padding и Margin – отступы**

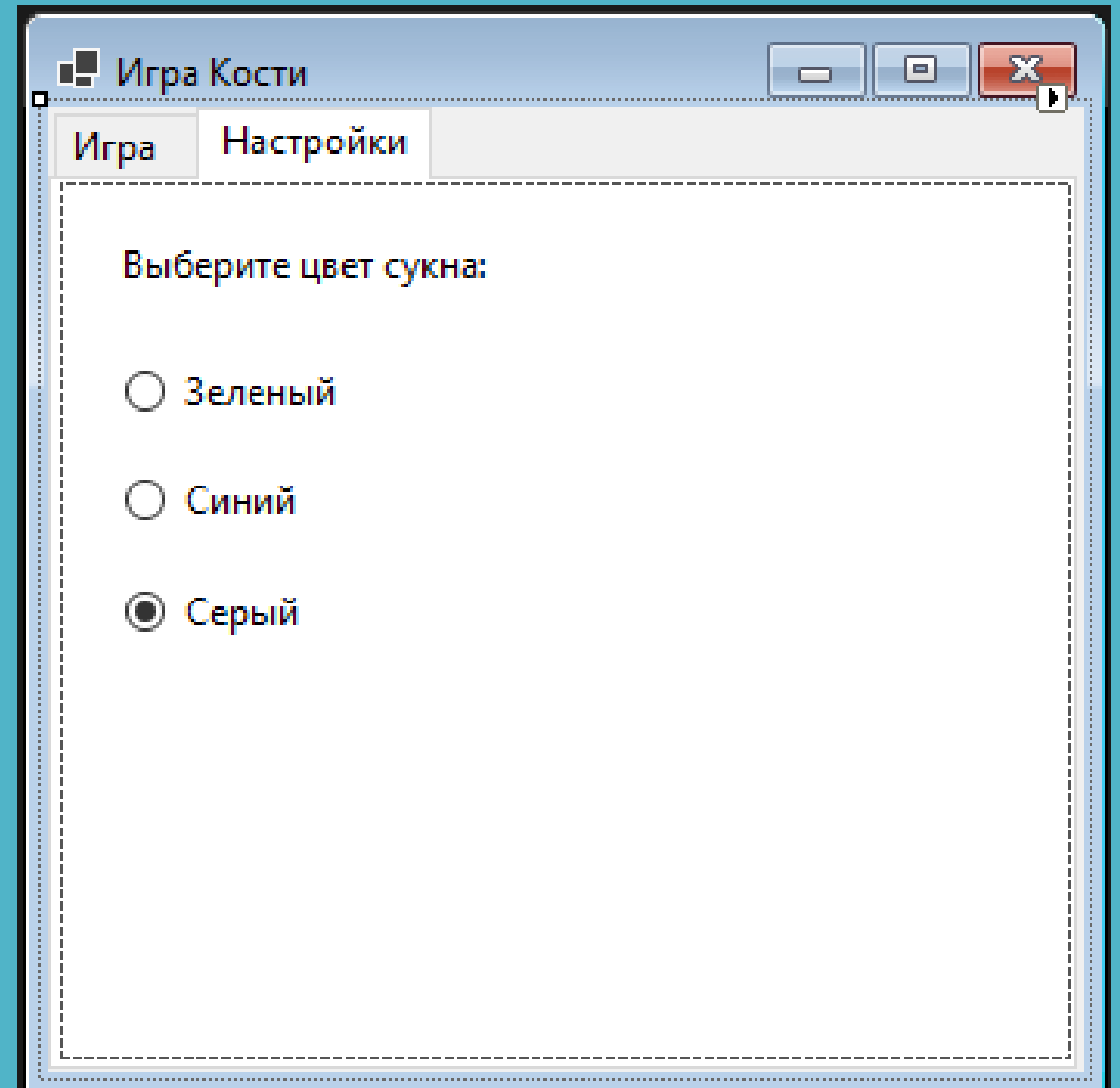
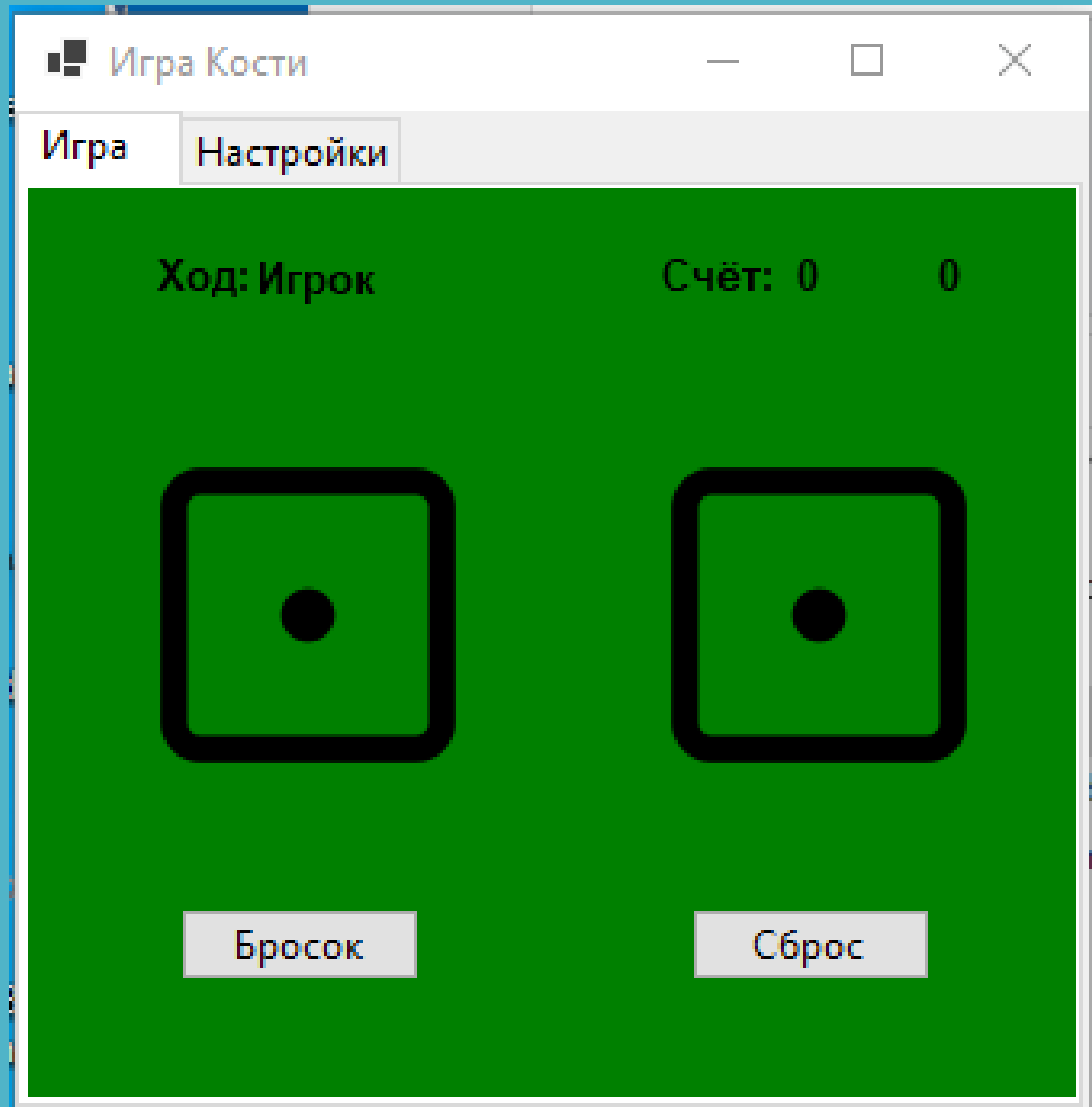
Padding – отступы внутри элемента управления (между границей элемента и его содержимым).

Margin – отступы вокруг элемента управления (между элементом и другими элементами или краями контейнера).

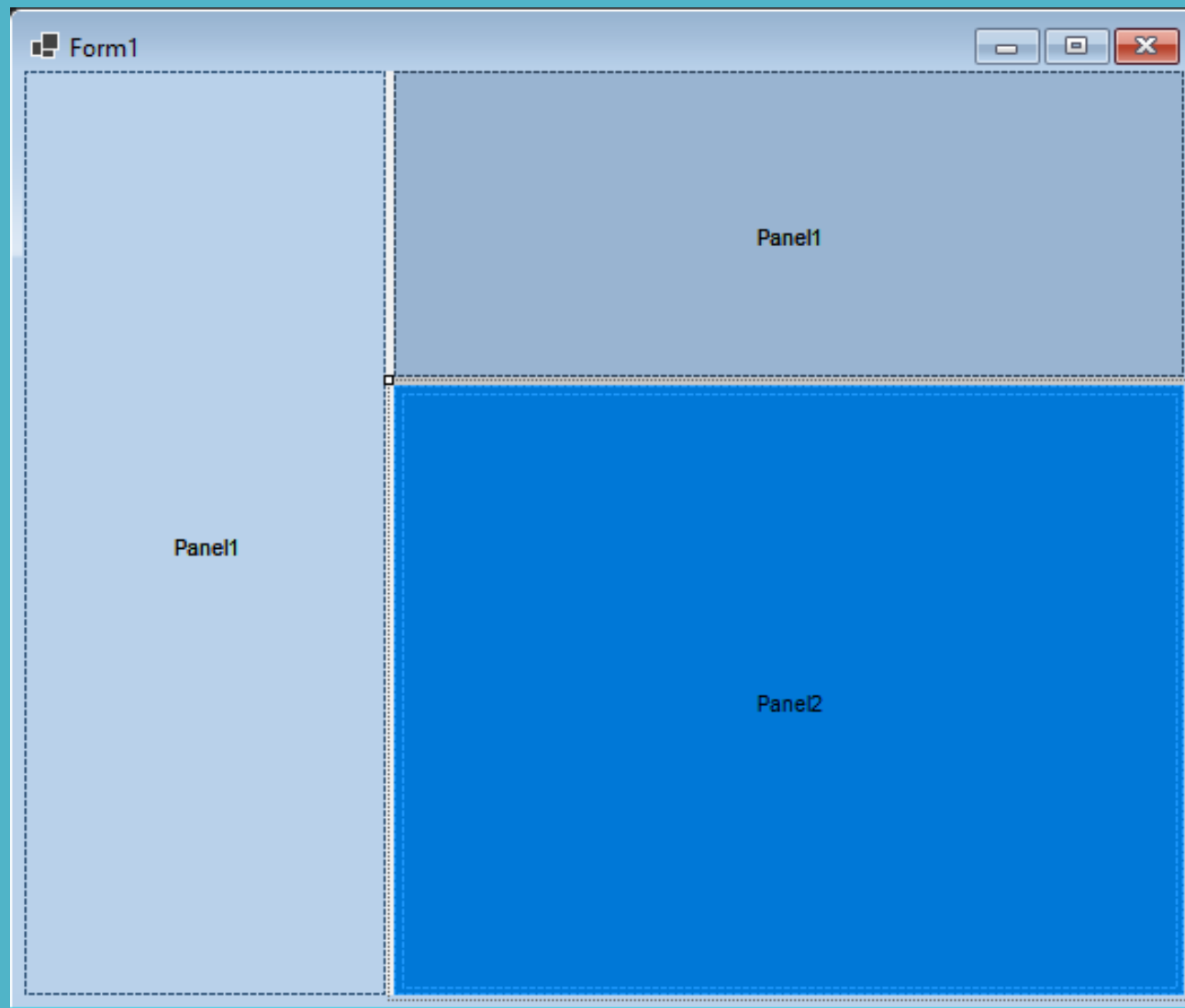
## 8. Примеры использования контейнеров.

- Создание форм с вкладками.
- Разделение формы на несколько областей.
- Автоматическое расположение элементов в потоке или таблице.

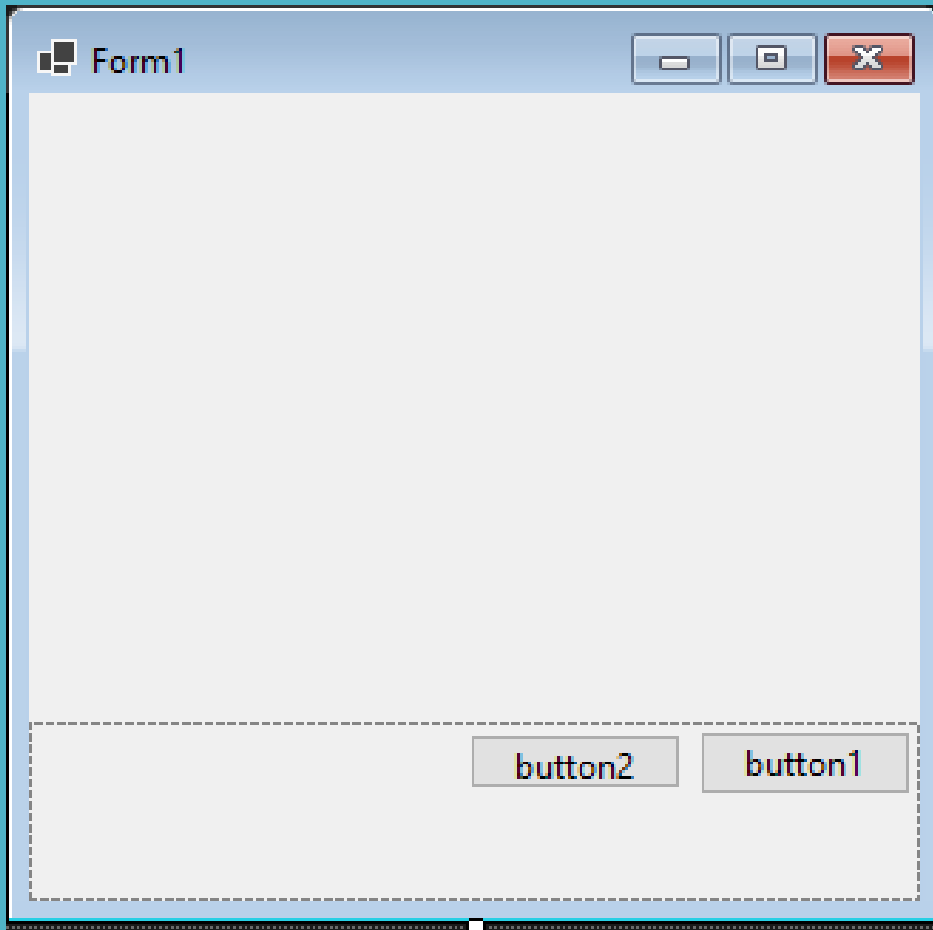
# Пример (TabControl)



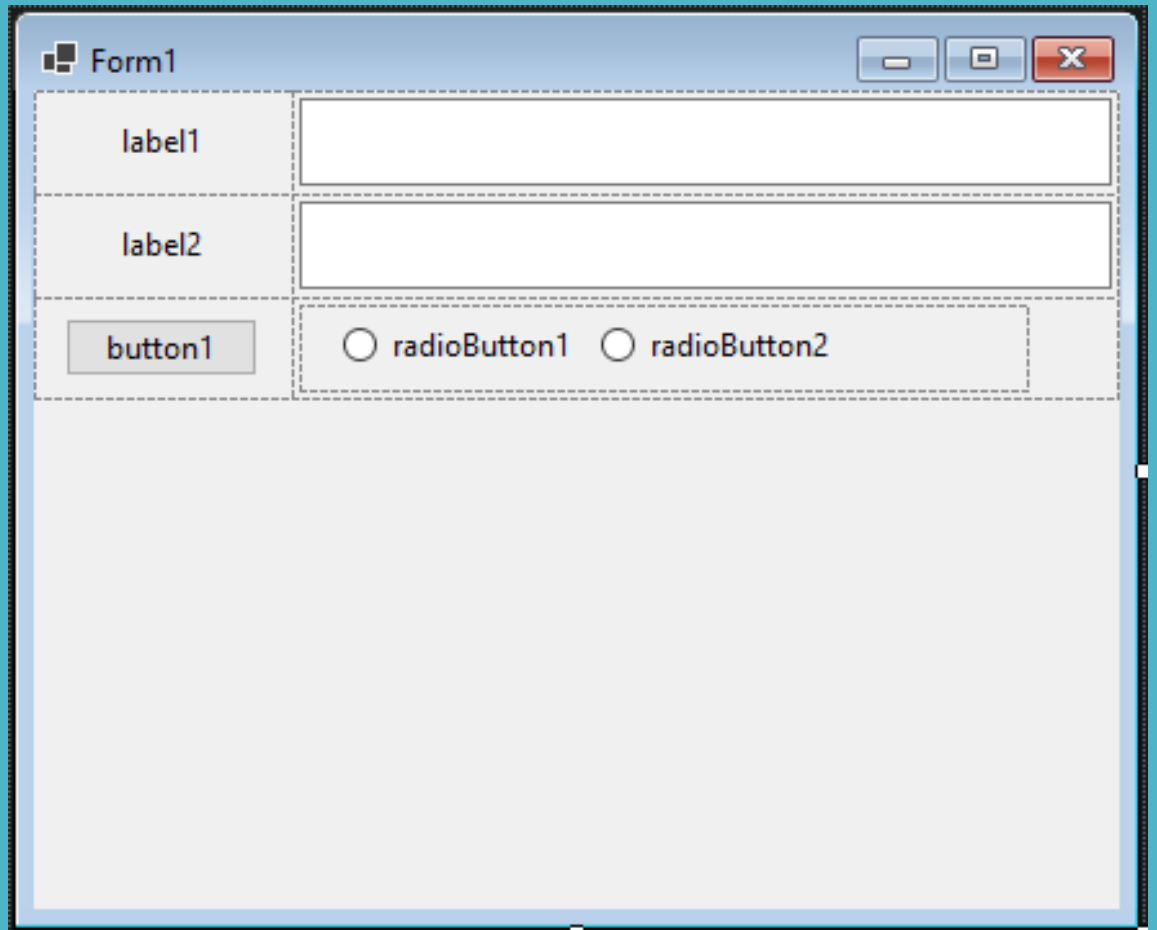
# Пример (SplitContainer)



# Пример (TableLayoutPanel и FlowLayoutPanel)



A screenshot of a Windows form titled "Form1". The form has a standard Windows title bar with minimize, maximize, and close buttons. The main area of the form is mostly empty. At the bottom, there is a dashed rectangular box representing a `FlowLayoutPanel`. Inside this box, two buttons are arranged horizontally: "button2" on the left and "button1" on the right.



A screenshot of a Windows form titled "Form1". The form has a standard Windows title bar. The top portion of the form is occupied by a dashed rectangular box representing a `TableLayoutPanel`. This panel is divided into three rows and two columns. The first row contains "label1" in the left column and an empty text box in the right column. The second row contains "label2" in the left column and another empty text box in the right column. The third row contains "button1" in the left column and two radio buttons, "radioButton1" and "radioButton2", in the right column. Below the `TableLayoutPanel`, the rest of the form is empty.

# Список литературы:

1. [Видеокурс C#.](#)
2. [Видеокурс C# Windows Forms](#)
3. [Metanit](#)
4. <https://metanit.com/sharp/windowsforms/3.1.php>
5. [Видеокурс C# Windows Forms](#)



# **Материалы лекций:**

<https://github.com/ShViktor72/Education>

# **Обратная связь:**

[colledge20education23@gmail.com](mailto:colledge20education23@gmail.com)

# Домашнее задание (Работа с элементами управления):

## Задание 1. Работа с CheckBox и RadioButton

Создайте Windows Forms-приложение с формой, на которой:

- Разместите три CheckBox, имитирующих выбор дополнительных услуг (например, "Доставка", "Подарочная упаковка", "СМС-уведомление").
- Разместите группу из трех RadioButton, позволяющих выбрать способ оплаты ("Наличные", "Карта", "Перевод").
- Добавьте Label, который будет отображать выбранные опции при нажатии на кнопку "Показать выбор".

Что нужно реализовать:

- При изменении состояния CheckBox обновлять список выбранных услуг.
- Гарантировать, что в группе RadioButton можно выбрать только один вариант.
- Выводить информацию о выборе в Label при нажатии на кнопку.

## Задание 2. Работа с ListBox и ComboBox

Создайте форму, на которой:

- Разместите ListBox для отображения списка товаров.
- Добавьте текстовое поле (TextBox) и кнопку "Добавить", чтобы пользователь мог добавлять товары в ListBox.
- Добавьте ComboBox с предустановленными категориями товаров (например, "Электроника", "Одежда", "Продукты").

Что нужно реализовать:

- При нажатии кнопки "Добавить" товар должен добавляться в ListBox.
- При выборе категории в ComboBox список товаров должен фильтроваться по выбранной категории.

### Задание 3. Работа с DateTimePicker и MonthCalendar

Создайте приложение для бронирования встречи, в котором:

- Разместите DateTimePicker для выбора даты и времени встречи.
- Разместите MonthCalendar, позволяющий выбрать диапазон дат.
- Добавьте Label, отображающий выбранную дату и время.
- Добавьте кнопку "Подтвердить", при нажатии на которую выбранные параметры будут выводиться в MessageBox.

Что нужно реализовать:

- Позволить выбрать дату и время встречи через DateTimePicker.
- Позволить выбрать диапазон дат через MonthCalendar.
- При нажатии на кнопку "Подтвердить" отображать MessageBox с выбранными значениями.

## Задание 4.

Создайте небольшую анкету с элементами управления:

- Имя (TextBox).
- Дата рождения (DateTimePicker).
- Пол (RadioButton).
- Увлечения (несколько CheckBox).
- Кнопку "Отправить", которая будет отображать введенные данные в MessageBox.

# Домашнее задание (Контейнеры):

Задание 1: Создание формы с использованием контейнеров

Создайте новое приложение Windows Forms в Visual Studio.

Используя контейнеры (Panel, GroupBox, TabControl, FlowLayoutPanel, TableLayoutPanel, SplitContainer), создайте форму, которая будет содержать следующие элементы:

Группировка элементов: Используйте GroupBox для группировки радиокнопок (например, выбор цвета фона: красный, синий, зеленый).

Вкладки: Используйте TabControl для создания двух вкладок. На первой вкладке разместите текстовое поле и кнопку, на второй — список (ListBox) с несколькими элементами.

Автоматическое расположение: Используйте FlowLayoutPanel для автоматического расположения нескольких кнопок (например, "Добавить", "Удалить", "Редактировать").

Табличное расположение: Используйте TableLayoutPanel для создания таблицы с двумя строками и двумя столбцами. В каждой ячейке разместите по одной кнопке.

Разделение области: Используйте SplitContainer для деления формы на две части. В левой части разместите TreeView, а в правой — TextBox.

## Задание 2: Управление расположением элементов

Используя свойства Dock, Anchor, AutoSize, Padding и Margin, настройте расположение элементов на форме:

Закрепите StatusStrip в нижней части формы с помощью свойства Dock.

Привяжите TextBox к верхнему и левому краям формы с помощью свойства Anchor.

Настройте автоматическое изменение размеров кнопки в зависимости от текста с помощью свойства AutoSize.

Установите отступы внутри Panel с помощью свойства Padding и отступы вокруг Button с помощью свойства Margin.

## Задание 3: Создание адаптивного интерфейса

Создайте форму, которая будет адаптироваться под изменение размеров окна:

Используйте `TableLayoutPanel` для создания сетки, которая будет растягиваться при изменении размеров окна.

Добавьте несколько элементов управления (например, кнопки, текстовые поля) в ячейки таблицы и настройте их свойства `Dock` и `Anchor` так, чтобы они корректно отображались при изменении размеров окна.



## Задание 4. Форма регистрации

Цель: Создать форму регистрации, используя TableLayoutPanel.

Требования:

Использовать TableLayoutPanel с 5 строками и 2 столбцами.

В ячейках разместить элементы управления:

Label "Имя" → TextBox

Label "Фамилия" → TextBox

Label "Email" → TextBox

Label "Пароль" → TextBox (с UseSystemPasswordChar = true)

Кнопка "Регистрация" (на всю ширину таблицы)

Сделать адаптивный интерфейс (при изменении размеров окна поля ввода должны растягиваться).

💡 Подсказка:

Используйте Dock = Fill для полей ввода.

Для кнопки "Регистрация" установите ColumnSpan = 2.

## Задание 5. Калькулятор

Цель: Создать калькулятор, используя `TableLayoutPanel` для кнопок.

Требования:

В верхней части формы разместить `TextBox` для вывода выражений.

Использовать `TableLayoutPanel` (4×4) для кнопок с числами (0-9) и операциями (+, -, \*, /, =, C).

Кнопки должны равномерно распределяться по ячейкам и изменять размеры при изменении окна.

Реализовать обработку нажатий кнопок.

 Подсказка:

Используйте `Anchor = Top, Bottom, Left, Right` для адаптивности.

Обработчик событий можно привязать ко всем кнопкам сразу.

## Задание 6. Панель инструментов

Цель: Создать окно с панелью инструментов, используя `FlowLayoutPanel`.

Требования:

В верхней части формы добавить `FlowLayoutPanel` с кнопками (Новый, Открыть, Сохранить, Выход).

Кнопки должны автоматически переноситься на новую строку, если окно сужается.

Добавить основную рабочую область (`Panel`), которая занимает оставшееся место.

При нажатии кнопок показывать `MessageBox` с соответствующим сообщением.

💡 Подсказка:

Установите `FlowDirection = LeftToRight`.

Свойство `WrapContents = true` обеспечит перенос кнопок.

## Задание 7. Файловый менеджер

Цель: Создать мини-файловый менеджер, используя SplitContainer.

Требования:

SplitContainer, разделяющий окно на 2 части:

Левая часть (ширина 30%) → TreeView для отображения структуры папок.

Правая часть (ширина 70%) → ListBox для списка файлов.

Добавить кнопки (Обновить, Удалить, Открыть), которые взаимодействуют с файлами.

Реализовать выбор папки в TreeView и отображение ее содержимого в ListBox.

💡 Подсказка:

Используйте Directory.GetFiles() и Directory.GetDirectories().

Для обновления содержимого обновляйте ListBox.Items.

## Задание 8. Настройки приложения (TabControl)

Цель: Создать окно настроек с вкладками.

Требования:

Использовать TabControl с тремя вкладками:

Общие (Checkbox "Запускать при старте", ComboBox "Тема оформления")

Сеть (TextBox "Адрес сервера", NumericUpDown "Порт")

О программе (Label с информацией о версии и авторе)

Все элементы должны сохраняться в Properties.Settings и загружаться при запуске.

Добавить кнопку "Сохранить", сохраняющую настройки.

💡 Подсказка:

Используйте `Properties.Settings.Default["ключ"] = значение;`

Вызовите `Properties.Settings.Default.Save();` при сохранении.

## Задание 9: Создание формы "Просмотр изображений"

Создайте форму для просмотра изображений:

SplitContainer для разделения формы на две части:

Слева — список изображений (ListBox).

Справа — область для отображения выбранного изображения (PictureBox).

Button для загрузки нового изображения (кнопка "Загрузить").

Button для удаления выбранного изображения (кнопка "Удалить").

Требования:

Используйте Dock для закрепления SplitContainer на всей форме.

Настройте автоматическое изменение размеров PictureBox в зависимости от размеров изображения с помощью свойства AutoSize.

Установите отступы вокруг кнопок с помощью свойства Margin.

Ответьте на вопросы:

- Какие контейнеры существуют в Windows Forms? Перечислите их и кратко опишите назначение каждого.
- В чем разница между Dock и Anchor? Приведите примеры их использования.
- Чем TableLayoutPanel отличается от FlowLayoutPanel? В каких случаях лучше использовать один из них?
- Что делает свойство AutoScroll? В каких контейнерах оно используется?
- Какие свойства используются для задания отступов у элементов? В чем разница между Margin и Padding?