МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образование «Полоцкий государственный университет имени Евфросинии Полоцкой»

Факультет информационных технологий

Кафедра технологий программирования

**Лабораторная работа №3**

по дисциплине: «Программирование на платформе .NET»

на тему: «Разработка gui. Создание sdi-приложений. Обработка событий»

Вариант №10

|  |  |
| --- | --- |
| Выполнил | Студент группы 21-ИТ-1  Чиникайло А.П. |
|  |  |
| Проверил | Преподаватель  Макарычева В.А. |

Полоцк, 2024

**ЦЕЛЬ РАБОТЫ**

Изучить принципы разработки графического интерфейса приложений для ОС Windows в Visual Studio .Net. Освоить использование элементов графического интерфейса для управления работой приложения. Освоить принципы построения иерархических меню, создания диалоговых окон. Изучить модель обработки событий в языке C#.

**ЗАДАНИЕ**

1. Создать SDI-приложение (Single Document Interface, однодокументный интерфейс) с элементами ввода и отображения полей класса из задания к лабораторной работе 2. Для этого используйте различные элементы управления: текстовые поля, списки, независимые и радиокнопки, а также панели и менеджеры компоновки.
2. Ввод новых данных осуществлять через дополнительную диалоговую форму.
3. При изменении данных запрашивать подтверждение через окно диалога. В случае неполных данных сообщать об ошибке.
4. Объекты сохранять в коллекции.
5. Реализовать просмотр всей коллекции объектов через список. Для редактирования выбранного объекта создать дополнительную форму модального диалога.
6. Добавить на форму меню, позволяющее работать с пунктами: добавить, просмотреть, удалить, редактировать, справка.
7. Дублировать основные операции панелью инструментов.

**ХОД РАБОТЫ**

Листинг 1 – Form1.cs

using lab\_3;

using lab\_3.Type;

using lab\_3.Info;

namespace lab\_3

{

// основная форма

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

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

{

Add add = new Add();

add.ShowDialog();

listShips.DataSource = Data.GetShips().ConvertAll(s => s.Name);

}

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

{

if (listShips.SelectedIndex == -1)

return;

Add add = new Add(listShips.SelectedIndex);

add.ShowDialog();

listShips.DataSource = Data.GetShips().ConvertAll(s => s.Name);

}

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

{

if (listShips.SelectedIndex == -1)

return;

Data.GetShips().RemoveAt(listShips.SelectedIndex);

listShips.DataSource = Data.GetShips().ConvertAll(s => s.Name);

}

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

{

if (listShips.SelectedIndex == -1)

{

Show show = new Show();

show.ShowDialog();

}

else

{

Show show = new Show(listShips.SelectedIndex);

show.ShowDialog();

}

}

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

{

addToolStripMenuAdd\_Click(sender, e);

}

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

{

editToolStripMenuItem\_Click(sender, e);

}

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

{

deleteToolStripMenuItem\_Click(sender, e);

}

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

{

showToolStripMenuItem\_Click(sender, e);

}

}

}

Листинг 2 – Add.cs

using lab\_3.Type;

using lab\_3.Info;

namespace lab\_3

{

// форма добавления и редактирования информации о кораблях

public partial class Add : Form

{

private ShipType shipType;

int? shipIndex;

public Add()

{

InitializeComponent();

}

public Add(int shipIndex) : this()

{

Ship ship = Data.GetShips()[shipIndex];

txbName.Text = ship.Name;

txbDisplacement.Text = ship.Displacement.ToString();

if (ship.getShipType() == ShipType.CargoShip)

{

RBCargoShip.Checked = true;

}

else if (ship.getShipType() == ShipType.PassengerShip)

{

RBPassengerShip.Checked = true;

}

else

{

RBIndustrialShip.Checked = true;

}

List<CabinCategory> cabinCategories = ship.getCabinCategories();

List<string> str = cabinCategories.ConvertAll(c => c.ToString());

for (int i = 0; i < checkedListBox.Items.Count; i++)

{

if (str.Contains(checkedListBox.Items[i].ToString()))

{

checkedListBox.SetItemChecked(i, true);

}

}

this.shipIndex = shipIndex;

}

public Add(int shipIndex, bool isShow) : this(shipIndex)

{

txbDisplacement.ReadOnly = isShow;

txbName.ReadOnly = isShow;

RBCargoShip.Enabled = isShow;

RBPassengerShip.Enabled = isShow;

RBIndustrialShip.Enabled = isShow;

checkedListBox.Enabled = isShow;

btnAccept.Enabled = isShow;

}

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

{

if (!CheckValues())

return;

var categories = Data.getCabinCategories();

Ship ship = new PassengerShip();

ship.Name = txbName.Text;

ship.Displacement = float.Parse(txbDisplacement.Text);

ship.setShipType(shipType);

List<CabinCategory> cabinCategories = new List<CabinCategory>();

foreach (var item in checkedListBox.CheckedItems)

{

cabinCategories.Add(categories[item.ToString()]);

}

ship.setCabinCategories(cabinCategories);

//Form1 form = new Form1();

//form.AddShip(ship);

if(!shipIndex.HasValue)

{

Data.GetShips().Add(ship);

}

else

{

DialogResult answer = MessageBox.Show("you sure that want to change this item?", "Question", MessageBoxButtons.YesNo, MessageBoxIcon.Question);

if(answer == DialogResult.Yes)

{

Data.GetShips()[shipIndex.Value] = ship;

}

else

{

return;

}

}

Close();

}

private bool CheckValues()

{

bool isValuesCorrect = true;

string message = string.Empty;

if (string.IsNullOrEmpty(txbName.Text))

{

isValuesCorrect = false;

message += "text box Name must not empty\n";

}

if (string.IsNullOrEmpty(txbDisplacement.Text))

{

isValuesCorrect = false;

message += "text box Displacement must not empty\n";

}

else

{

if (!float.TryParse(txbDisplacement.Text, out float value))

{

isValuesCorrect = false;

message += "text box Displacement must be number\n";

}

}

if (!RBCargoShip.Checked && !RBPassengerShip.Checked && !RBIndustrialShip.Checked)

{

isValuesCorrect = false;

message += "you need to check any ship type\n";

}

else

{

}

if (checkedListBox.CheckedItems.Count == 0)

{

isValuesCorrect = false;

message += "you need to check any cabin categories\n";

}

if (!isValuesCorrect)

{

MessageBox.Show(message);

}

return isValuesCorrect;

}

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

{

shipType = ShipType.CargoShip;

}

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

{

shipType = ShipType.PassengerShip;

}

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

{

shipType = ShipType.IndustrialShip;

}

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

{

Close();

}

private void Add\_Load(object sender, EventArgs e)

{

}

}

}

Листинг 3 – Show.cs

using lab\_3.Info;

using lab\_3.Type;

namespace lab\_3

{

// форма просмотра информации о кораблях

public partial class Show : Form

{

private const string EMPTY = "Empty";

public Show()

{

InitializeComponent();

List<Ship> shipList = Data.GetShips();

if(shipList.Count == 0)

{

ShowEmpty();

}

else

{

CBShipNames.DataSource = shipList.ConvertAll(s => s.Name);

CBShipNames.SelectedIndex = 0;

}

}

public Show(int shipIndex) : this()

{

//InitializeComponent();

ShowInfoShip(shipIndex);

CBShipNames.SelectedIndex = shipIndex;

}

private void ShowInfoShip(int shipIndex)

{

labCabinCategories.Text = string.Empty;

Ship ship = Data.GetShips()[shipIndex];

labName.Text = ship.Name;

labDisplacement.Text = ship.Displacement.ToString();

labShipType.Text = ship.getShipType().ToString();

List<CabinCategory> cabinCategories = ship.getCabinCategories();

for (int i = 0; i < cabinCategories.Count; i++)

{

labCabinCategories.Text += cabinCategories[i].ToString() + Environment.NewLine;

}

}

public void ShowEmpty()

{

labName.Text = EMPTY;

labDisplacement.Text = EMPTY;

labShipType.Text = EMPTY;

labCabinCategories.Text = EMPTY;

}

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

{

Close();

}

private void label5\_Click(object sender, EventArgs e)

{

}

private void CBShipNames\_SelectedIndexChanged(object sender, EventArgs e)

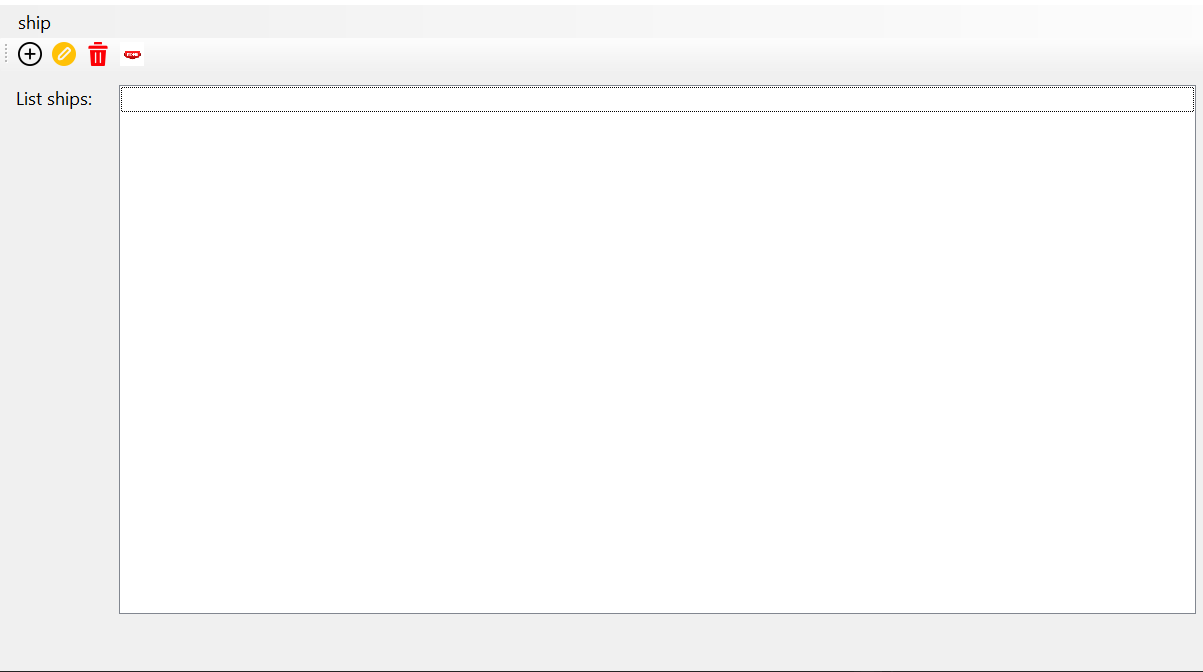
{

ShowInfoShip(CBShipNames.SelectedIndex);

}

}

}

Рисунок 1 – Главное окно программы

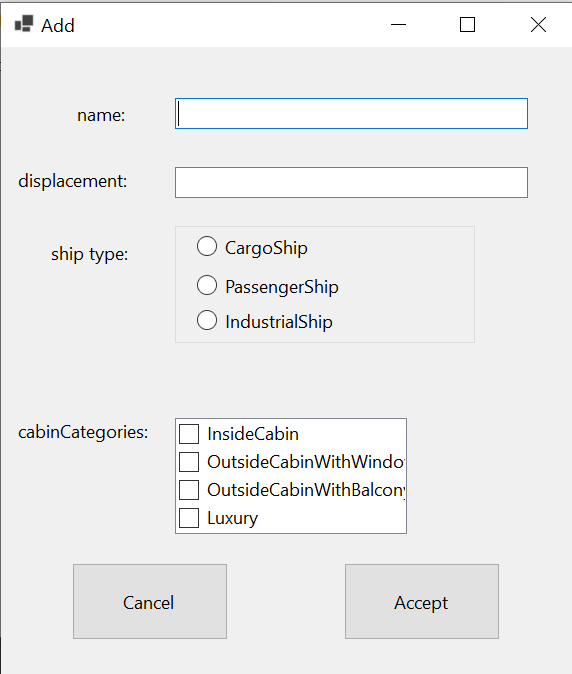


Рисунок 2 – Окно добавления нового корабля

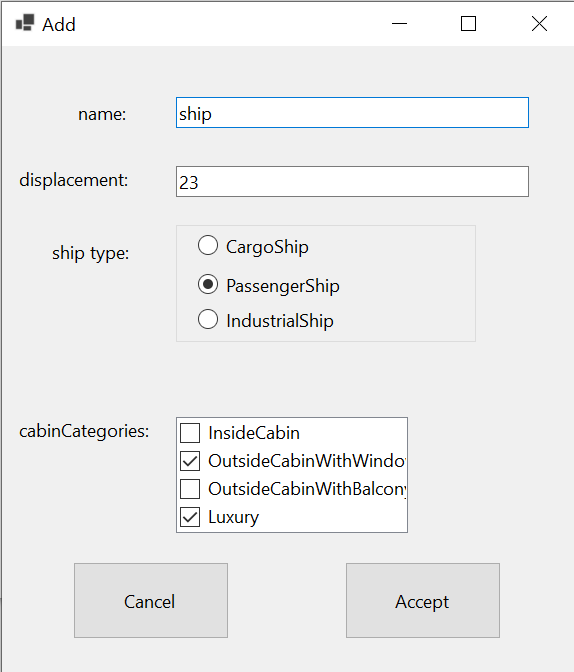


Рисунок 3 – Окно изменения информации о корабле

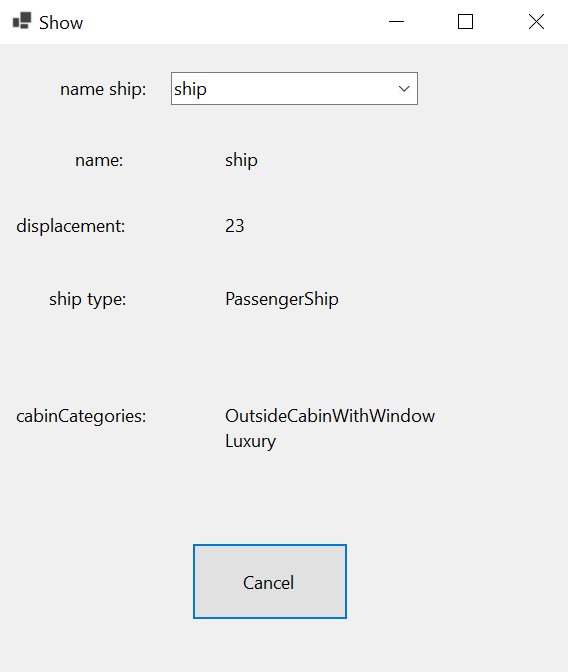


Рисунок 4 – Окно просмотра информации о корабле

**ОТВЕТЫ НА ВОПРОСЫ**

1. Что такое форма и ее назначение?

Форма — это окно приложения, которое отображает элементы управления и обеспечивает взаимодействие с пользователем. Она служит основой графического интерфейса программы и отвечает за отображение и управление содержимым приложения.

1. Что такое элементы управления? На какие группы они делятся?

Элементы управления — это компоненты интерфейса, которые обеспечивают взаимодействие пользователя с программой. Основные группы:

* Редактирование текста (TextBox, RichTextBox).
* Отображение текста (Label, StatusBar).
* Выбор из списка (ComboBox, ListBox).
* Ввод значений (CheckBox, RadioButton).
* Отображение графики (PictureBox).
* Диалоговые панели (OpenFileDialog, SaveFileDialog).
* Команды (Button, ToolBar).
* Создание меню (MenuStrip).

1. Как установить элемент управления на форму и задать его свойства?

Элемент управления можно перетащить на форму из панели инструментов (Toolbox). Его свойства задаются через окно свойств (Properties), где можно изменить текст, размер, цвет и другие параметры.

1. Что такое событие? Как в .Net реализуются события?

Событие — это действие, которое может произойти в программе (например, нажатие кнопки или перемещение мыши). В .Net события реализуются через механизм делегатов. Событие связывается с обработчиком, который выполнит код в ответ на событие.

1. Что такое обработчик события?

Обработчик события — это метод, который вызывается в ответ на событие. Он содержит код, который будет выполнен при возникновении определенного действия (например, щелчок по кнопке).

1. Как выбрать событие для элемента управления?

Для выбора события нужно открыть окно "События" (вкладка с иконкой молнии в Properties) и выбрать нужное событие. Двойной щелчок на событии создаст его обработчик.

1. Что такое делегат?

Делегат — это тип, который указывает на метод с определенной сигнатурой. Делегаты используются для связи событий с обработчиками.

1. Как задаются обработчики событий для элементов управления?

Обработчики задаются через привязку делегата к событию с использованием оператора `+=`. Например:

this.Button1.Click += new EventHandler(this.Button1\_Click);

1. Как происходит подключение к прослушиванию события?

Подключение происходит через создание делегата и добавление его к событию. Пример:

this.Click += new EventHandler(MethodName);

1. Как создать собственные события и их обработчики?

Собственные события создаются через ключевое слово `event` и делегаты. Для создания обработчика используется стандартная структура:

public event EventHandler MyEvent;

protected virtual void OnMyEvent(EventArgs e) {

MyEvent?.Invoke(this, e);

}

1. Как создать верхнее меню?

Используется компонент MenuStrip. Перетащите его на форму, добавьте пункты меню через редактор, и задайте им события.

1. Как добавить выпадающее меню в верхнее меню?

Для этого в MenuStrip добавляются пункты меню, внутри которых можно создавать дочерние элементы, которые будут являться выпадающими пунктами.

1. Как установить определенному пункту меню сочетание клавиш?

В окне свойств элемента меню в свойстве `ShortcutKeys` можно выбрать нужное сочетание клавиш.

1. Как создать панель инструментов?

Перетащите компонент ToolStrip на форму из панели инструментов. Добавьте кнопки или другие элементы на панель.

1. Как добавить несколько кнопок на панель инструментов?

На панели ToolStrip нажмите правой кнопкой мыши и выберите "Добавить кнопку". Повторите это действие для нескольких кнопок.

1. Как разместить на кнопке изображение?

В свойстве `Image` кнопки выберите изображение для отображения на кнопке. Используйте `Set Image` для загрузки изображения из файла.

1. Как добавить новую форму в приложение?

В Visual Studio выберите "Проект" → "Добавить форму", затем выберите "Windows Forms". Форма будет добавлена в проект.

1. Как организовать переход к добавленной форме?

Переход к новой форме осуществляется с помощью создания экземпляра этой формы и вызова метода `Show()` или `ShowDialog()`:

Form2 form2 = new Form2();

form2.Show();

1. Что такое модальная форма и немодальная? Как они вызываются?

Модальная форма блокирует взаимодействие с основной формой до закрытия. Вызывается через `ShowDialog()`. Немодальная форма позволяет работать с другими окнами и вызывается через `Show()`.

1. Как организовать передачу данных между формами?

Для передачи данных между формами используются свойства или конструкторы форм. Например, в форме передается значение через свойство:

form2.SomeProperty = value;

1. Как добавить новый класс в проект?

В Visual Studio выберите "Проект" → "Добавить класс", введите имя класса и нажмите "Добавить".

**ВЫВОД**

В результате выполнения лабораторной работы я изучил принципы разработки графического интерфейса приложений для ОС Windows в Visual Studio .Net и модель обработки событий в языке C#, а также освоил использование элементов графического интерфейса для управления работой приложения и принципы построения иерархических меню, создания диалоговых окон.