МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное

образовательное учреждение высшего образования

«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

ФАКУЛЬТЕТ АВТОМАТИКИ И ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ

Кафедра Автоматизированных систем управления

**ОТЧЕТ О ВЫПОЛНЕНИИ**

**Лабораторной работы № 2**

Дисциплина «*Технологии и методы программирования*»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Выполнили:** |  | **Проверил:** |  | |
| Студенты | *Бузмаков А.И.,*  *Шестаков К.Д.* | Преподаватель | *Астапчук В. А., Пустовских Д. А.* | |
| Факультет | *АВТФ* |  |  | |
| Направление (специальность) подготовки | *09.03.01 – Информатика и вычислительная техника* | Балл: \_\_\_\_\_\_\_\_\_ | |  |
| Группа | *АП-227* | Оценка *\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_* | | |
| Шифр |  |  |  | |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | | |
| подпись | | подпись | | |
| Дата сдачи: | «1» апреля 2024г. | Дата защиты: | «\_\_»\_\_\_\_\_\_\_\_\_\_20\_\_г. | |

Новосибирск 2024

**Текст задания**

1. Реализовать класс, конструктор которого создаёт меню, структура которого определяется данными, находящимися во внешнем текстовом файле. Аргументом конструктора будет имя файла (по умолчанию, menu.txt). Меню программы рассматривается как множество деревьев (иерархий), корнями которых являются пункты главного меню (уровень иерархии 0). Следующие уровни иерархии представляют подпункты, при этом количество уровней произвольное для каждого пункта (подпункта). Каждая запись файла меню имеет следующую структуру:

Номер\_уровня\_в\_иерархииПробелНазвание\_пунктаПробелСтатус\_пунктаПробелИмяМетода

Последовательность записей соответствует последовательности пунктов. Статус пункта определяется:

0 – пункт виден и доступен;

1 – пункт виден, но не доступен;

2 – пункт не виден.

ИмяМетода – имя обработчика клика мыши на пункте меню.

Если имя обработчика не указано, это означает, что пункт имеет подменю, которое должно появиться после выбора этого пункта.

1. Внешний вид меню бизнес-приложений (предоставляемая функциональность), как правило, определяется ролью пользователя (его правами на доступ к функциям и данным). Список возможных пользователей будет храниться в файле USERS, записи которого имеют следующую структуру.

#Имя\_пользователяПробелПароль

Название\_пунктаПробелСтатус\_пункта\_для этого пользователя

Другие пункты, доступные пользователю

#Имя\_пользователяПробелПароль

Название\_пунктаПробелСтатус\_пункта\_для этого пользователя

Другие пункты, доступные пользователю

По умолчанию, отсутствующий в списке пункт имеет статус 0 (виден и доступен).

3) Любым текстовым редактором создать требуемые файлы. По умолчанию расширение имени файлов **.txt** .

4) Написать тестовую программу (оконное приложение) для проверки работы класса. При запуске программа открывает окно авторизации (см. рис. 1). Внешний вид окна должен быть как на рисунке. Имя приложения может быть произвольным. Номер версии приложения является параметром проекта.

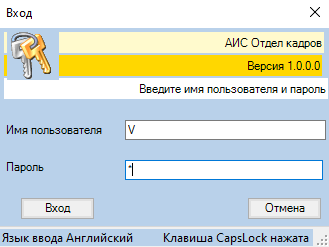


Рисунок 1 – Окно авторизации пользователя

Если авторизация прошла успешно, открывается главное окно программы с меню. (см. рис.2), сформированным для этого пользователя.

5) При выборе мышью пункта срабатывает обработчик пункта или появляется подменю, если имя метода не указано. Обработчик пункта должен вызвать метод с указанным именем, который просто обозначит себя.

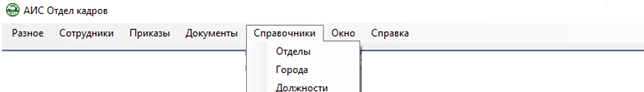


Рисунок 2 - Пример меню оконного приложения

Файл, содержащий описание этого меню, содержит следующие записи:

0 Разное 0 Others

0 Сотрудники 0 Stuff

0 Приказы 0 Orders

0 Документы 0 Docs

0 Справочники 0

1 Отделы 0 Departs

1 Города 0 Towns

1 Должности 0 Posts

0 Окно 0 Window

0 Справка 0

1 Оглавление 0 Content

1 О программе 0 About

Для этого примера список пользователей содержит:

Имя – Boss. Он имеет доступ ко всем функциям (пунктам меню), кроме Справочники.

HR – инспектор отдела кадров. Он имеет полный доступ к функциям Сотрудники, Приказы и Справочники.

OO – сотрудник общего отдела. Он имеет полный доступ к функции Документы и Справочники.

Пункты Разное, Окно и Справка (Оглавление, О программе) доступны всем.

**Тексты разработанных программ**

|  |
| --- |
| MainForm.cs |
| using System;  using System.Collections.Generic;  using System.IO;  using System.Windows.Forms;  namespace laba2  {  public partial class MainForm : Form  {  private string \_username;  public MainForm(string username)  {  this.\_username = username;  InitializeComponent();  LoadUserRoles(\_username);  }  // Метод для загрузки ролей пользователя из файла  private void LoadUserRoles(string username)  {  // Словарь для хранения ролей пользователя  Dictionary<string, string> userRoles = new Dictionary<string, string>();  // Чтение строк из файла пользователей  string[] lines = File.ReadAllLines("users.txt");  // Перебор строк файла  for (int i = 0; i < lines.Length; i++)  {  string[] parts = lines[i].Split(' ');  if ("#" + username == parts[0])  {  for (int j = i + 1; j < lines.Length; j++)  {  if (!lines[j].StartsWith("#"))  {  string[] roleParts = lines[j].Split(' ');  string name = "";  string status = "";  for (int k = 0; k < roleParts.Length; k++)  {  bool isNumeric = int.TryParse(roleParts[k], out \_);  if (!isNumeric)  {  name += roleParts[k] + " ";  }  else  {  status = roleParts[k];  }  }  name = name.Trim();  status = status.Trim();  userRoles[name] = status;  }  else  {  break;  }  }  break;  }  }  LoadMenu("menu.txt", menuStrip1, userRoles);  }  // Метод для загрузки меню из файла  private void LoadMenu(string fileName, MenuStrip menuStrip, Dictionary<string, string> userRoles)  {  if (File.Exists(fileName))  {  string[] lines = File.ReadAllLines(fileName);  Dictionary<int, List<ToolStripMenuItem>> menuItemsByLevel = new Dictionary<int, List<ToolStripMenuItem>>();  foreach (string line in lines)  {  string[] parts = line.Split(' ');  if (parts.Length >= 4)  {  int level = int.Parse(parts[0]);  string title = "";  int status = 0;  string methodName = "";  for (int i = 1; i < parts.Length; i++)  {  bool isNumeric = int.TryParse(parts[i], out \_);  if (!isNumeric)  {  title += parts[i] + " ";  }  else  {  status = int.Parse(parts[i]);  methodName = parts[i + 1];  break;  }  }  title = title.Trim();  if (userRoles.ContainsKey(title))  {  status = Int32.Parse(userRoles[title]);  }  ToolStripMenuItem menuItem = new ToolStripMenuItem(title, null, MenuItem\_Click);  menuItem.Tag = methodName;  if (status == 2)  {  menuItem.Visible = false;  }  else if (status == 1)  {  menuItem.Enabled = false;  }  if (!menuItemsByLevel.ContainsKey(level))  {  menuItemsByLevel[level] = new List<ToolStripMenuItem>();  }  menuItemsByLevel[level].Add(menuItem);  if (level == 0)  {  menuStrip.Items.Add(menuItem);  }  else  {  int parentLevel = level - 1;  if (menuItemsByLevel.ContainsKey(parentLevel))  {  foreach (var parentMenuItem in menuItemsByLevel[parentLevel])  {  parentMenuItem.DropDownItems.Add(menuItem);  }  }  }  }  }  }  }  // Метод для поиска родительского пункта меню по уровню  private ToolStripMenuItem FindParentMenuItem(ToolStripItemCollection items, int level)  {  foreach (ToolStripMenuItem item in items)  {  if (item.Tag != null && (int)item.Tag == level)  {  return item;  }  if (item.DropDownItems.Count > 0)  {  ToolStripMenuItem foundItem = FindParentMenuItem(item.DropDownItems, level);  if (foundItem != null)  return foundItem;  }  }  return null;  }  // Обработчик события нажатия на элемент меню  private void MenuItem\_Click(object sender, EventArgs e)  {  ToolStripMenuItem menuItem = (ToolStripMenuItem)sender;  string methodName = menuItem.Tag as string;  if (!string.IsNullOrEmpty(methodName))  {  InvokeMethodByName(methodName);  }  }  // Метод для вызова метода по имени  private void InvokeMethodByName(string methodName)  {  MessageBox.Show($"{methodName}");  }  }  } |

|  |
| --- |
| AuthForm.cs |
| using System;  using System.Collections.Generic;  using System.IO;  using System.Windows.Forms;  namespace laba2  {  public partial class AuthForm : Form  {  public AuthForm()  {  InitializeComponent();  }  // Обработчик нажатия кнопки "Отмена"  private void cancelButton\_Click(object sender, EventArgs e)  {  Application.Exit();  }  // Обработчик нажатия кнопки "Вход"  private void enterButton\_Click(object sender, EventArgs e)  {  // Чтение всех строк из файла пользователей  string[] lines = File.ReadAllLines("users.txt");  // Создание словаря для хранения пар логин-пароль  Dictionary<string, string> users = new Dictionary<string, string>();  // Перебор каждой строки файла  foreach (string line in lines)  {  string[] parts = line.Split(' ');  // Проверка, является ли строка логином и паролем  if (line.StartsWith("#"))  {  // Добавление пары логин-пароль в словарь  users[parts[0]] = parts[1].Trim();  }  }  // Проверка наличия введенного логина в словаре  if (users.ContainsKey("#" + loginBox.Text))  {  // Проверка правильности введенного пароля  if (passBox.Text == users["#" + loginBox.Text])  {  MessageBox.Show("Успешно!"); // Оповещение об успешном входе  MainForm mainForm = new MainForm(loginBox.Text); // Создание главной формы с передачей логина  mainForm.Show(); // Отображение главной формы  }  else  {  MessageBox.Show("Неправильный пароль!"); // Оповещение о неправильном пароле  }  }  else  {  MessageBox.Show("Неправильный логин!"); // Оповещение о неправильном логине  }  }  }  } |

**Описание методов**

### Класс AuthForm:

Этот класс представляет форму аутентификации пользователя.

#### Методы:

1. AuthForm(): Конструктор класса AuthForm, инициализирует компоненты формы.
2. cancelButton\_Click(object sender, EventArgs e): Обработчик события нажатия кнопки "Отмена", который завершает приложение.
3. enterButton\_Click(object sender, EventArgs e): Обработчик события нажатия кнопки "Вход", который производит аутентификацию пользователя по данным, введенным в текстовые поля. Проверяет введенный логин и пароль, сравнивая их с данными из файла пользователей. Если введенные данные верны, отображает главную форму (MainForm), передавая в неё логин пользователя. В случае неправильных данных выдает соответствующее сообщение.

### Класс MainForm:

Этот класс представляет главную форму приложения.

#### Поля:

1. \_username: Приватное поле для хранения имени пользователя.

#### Методы:

1. MainForm(string username): Конструктор класса MainForm, инициализирует главную форму и загружает роли пользователя, переданные через параметр username.
2. LoadUserRoles(string username): Метод для загрузки ролей пользователя из файла. Он читает строки из файла users.txt, находит роли для указанного пользователя и загружает их в словарь userRoles.
3. LoadMenu(string fileName, MenuStrip menuStrip, Dictionary<string, string> userRoles): Метод для загрузки меню из файла. Он читает строки из указанного файла, строит меню на основе уровня доступа пользователя (определенного в userRoles) и добавляет его к указанному menuStrip.
4. FindParentMenuItem(ToolStripItemCollection items, int level): Метод для поиска родительского пункта меню по уровню. Он рекурсивно обходит все пункты меню и ищет пункт с заданным уровнем.
5. MenuItem\_Click(object sender, EventArgs e): Обработчик события нажатия на элемент меню. Он вызывает соответствующий метод на основе имени метода, хранящегося в свойстве Tag элемента меню.
6. InvokeMethodByName(string methodName): Метод для вызова метода по его имени. Он отображает сообщение с именем вызываемого метода в случае успешного вызова.

**Тексты управляющих файлов**

|  |
| --- |
| users.txt |
| #Boss Boss  Справочники 2  #HR HR  Сотрудники 0  Приказы 0  Справочники 0  #OO OO  Документы 0  Справочники 0 |

|  |
| --- |
| menu.txt |
| 0 Разное 0 Others  0 Сотрудники 0 Stuff  0 Приказы 0 Orders  0 Документы 0 Docs  0 Справочники 0  1 Отделы 0 Departs  1 Города 0 Towns  1 Должности 0 Posts  0 Окно 0 Window  0 Справка 0  1 Оглавление 0 Content  1 О программе 0 About |

**Скриншоты приложения для каждого пользователя**

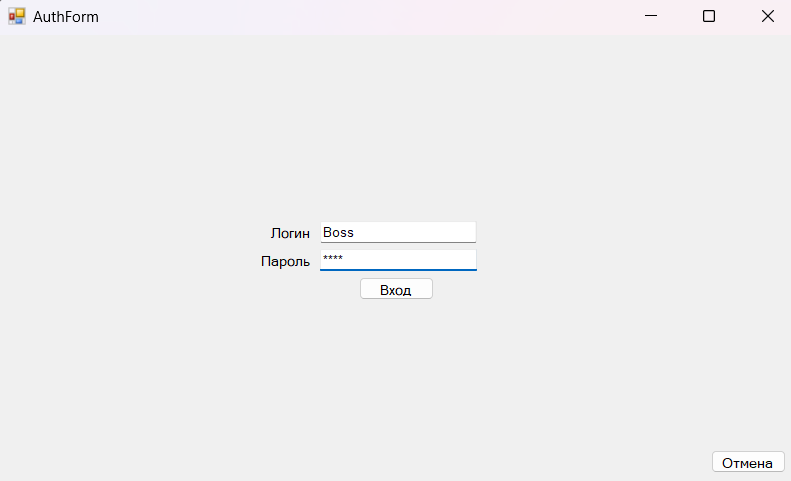


Рисунок 3 – Окно авторизации

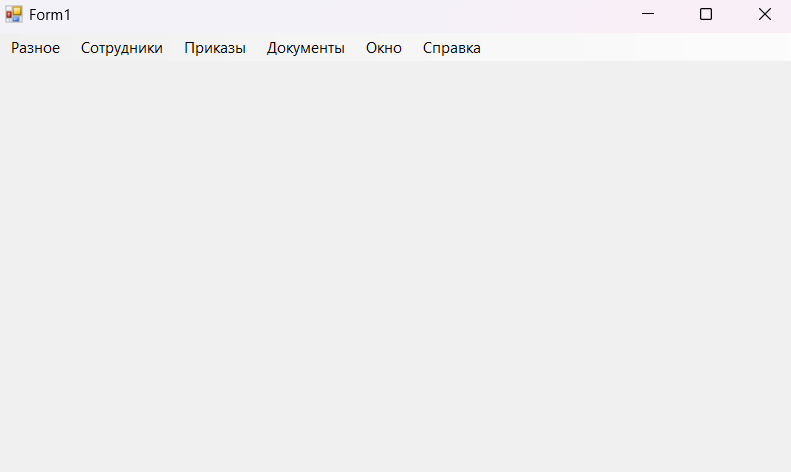


Рисунок 4 – Окно для пользователя Boss

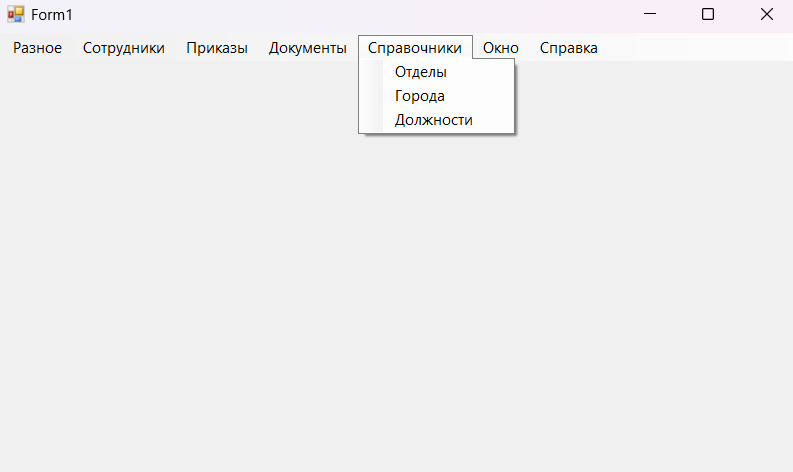


Рисунок 5 – Окно для пользователя HR

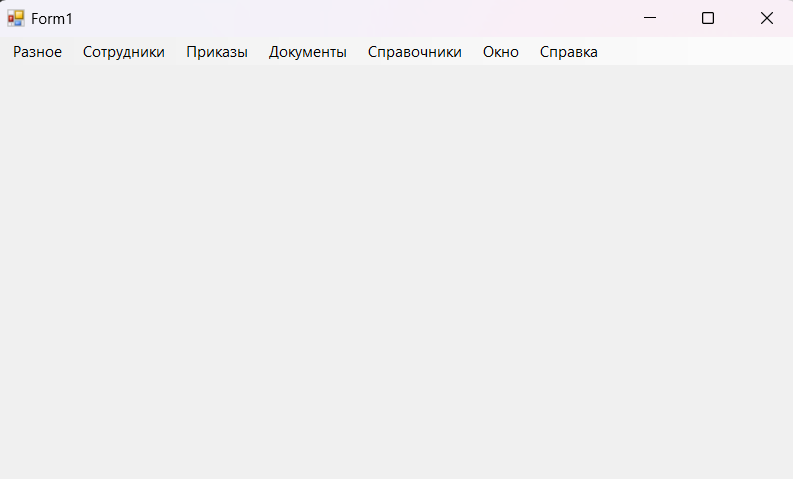


Рисунок 6 – Окно для пользователя OO