**ЛАБОРАТОРНА РОБОТА № 4**

***Системний реєстр***

**Хід роботи:**

***Мета:*** Вивчити призначення та методи роботи з системним реєстром Windows.

Посилання на репозиторій:

<https://github.com/andreylion06/SystemNetworkProgramming-Lab4>

***Завдання***:

1. Вивчити призначення та структуру системного реєстру.

2. Вивчити призначення та методи роботи з утилітою RegEdit.

3. Знайти відповідні розділи реєстру в яких є інформація про програми та служби які завантажуються автоматично.

4. Програмно, вивести список усіх програм та служб які завантажуються автоматично для усіх користувачів та поточного користувача.

5. Додати програмно до автозавантаження програм для поточного користувача завантаження програми WinWord або іншої.

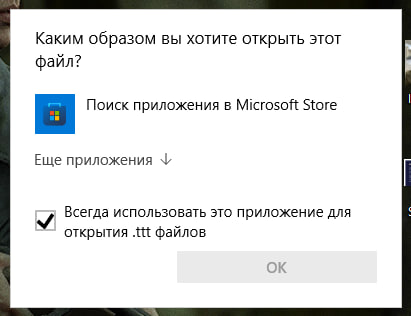
6. Вивести список повторно, та показали що зареєстрована програма є у списку.

7. Вивести список усіх завдань, які зареєстровані у планувальнику задач системи. Інформації отримати з відповідного розділу реєстру, як для усіх користувачів так і для поточного користувача.

8. Зробити програмно копію будь якого розділу реєстру у файл відповідного формату .reg

9. За допомогою текстового редактора створити REG файл, за допомогою якого в реєстр у відповідний розділ буде внесено інформацію про асоціацію відкриття файлів .ttt програмою notepad.

Створюю файл testreg.ttt та спробую його відкрити, маю діалогове вікно:



Створюю REG файл «association.ttt.reg» з наступним лістингом:  
  
Windows Registry Editor Version 5.00

[HKEY\_CLASSES\_ROOT\.ttt]

@="tttfile"

[HKEY\_CLASSES\_ROOT\tttfile]

@="Notepad Document"

[HKEY\_CLASSES\_ROOT\tttfile\DefaultIcon]

@="\"notepad.exe\",0"

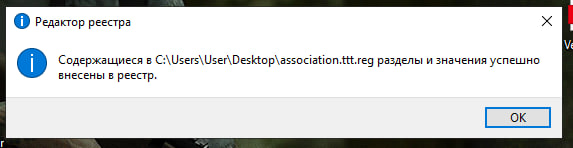
[HKEY\_CLASSES\_ROOT\tttfile\shell]

[HKEY\_CLASSES\_ROOT\tttfile\shell\open]

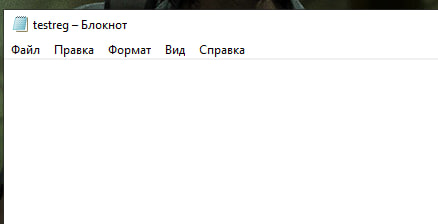
[HKEY\_CLASSES\_ROOT\tttfile\shell\open\command]

@="\"notepad.exe\" \"%1\""

Після збереження запускаю файл та маю діалоговий резульатат про те, що асоціаціюуспішно внесено в реєстр.



Повторюю відкриття файлу testreg.ttt, відкриття відбулось автоматично у Notepad:



10. Описати усі виконані дії у звіті.

***Звіт про програму:***

Програма "ConsoleAppLab4" призначена для управління автозавантаженням програм та відслідковування задач у планувальнику системи. Ця програма надає користувачеві доступ до різноманітної інформації про автозавантаження та планувальник задач, а також забезпечує можливість додавати нові програми до списку автозавантаження.

Рішення реалізоване за допомогою консольного додатку використовуючи мову програмування C#. Консольний додаток має меню з опціями, що описане класом Menu. Кожен пункт меню в класі – це, по суті, елемент словника, де ключ – це назва опції типу string та значення це делегат Action.

Структура програми:

* Program.cs: Включає в себе основний код програми, створює екземпляри класів RegManagerTest, TaskServiceManager та Menu, та обробляє основний цикл вибору опцій меню.
* Menu: Відображає консольне меню опцій, обробляє вибір користувача та виконує відповідні дії.
* RegManager: Надає методи для взаємодії з реєстром Windows, зокрема для отримання списку програм, що автоматично завантажуються, та додавання програм до списку автозавантаження.
* TaskServiceManager: Забезпечує інформацію про задачі, зареєстровані у планувальнику системи.
* RegManagerTest: Використовується для тестування функціоналу класу RegManager та передає методи як делегати у меню.

***Лістинг програми:***

using ConsoleAppLab4.Logic;

// Ім'я файлу: Program.cs

// Ремарка: Program.cs використовує класи Menu, RegManagerTest,

// TaskServiceManager та демонструє правильність їх роботи

// Автор: Андрій Сахно

var regManagerTest = new RegManagerTest();

var taskServiceManager = new TaskServiceManager();

var items = new Dictionary<string, Action>

{

{ "Listing startup programs for all users", () => regManagerTest.ListStartupProgramsAllUsers() },

{ "Listing startup programs for current user", () => regManagerTest.ListStartupProgramsCurrentUser() },

{ "Add Spotify to current user startup", () => regManagerTest.AddSpotifyToCurrentUserStartup() },

{ "Display all user tasks", () => taskServiceManager.DisplayAllUserTasks() },

{ "Display current user tasks", () => taskServiceManager.DisplayCurrentUserTasks() },

{ "Make copy of test registry key", () => regManagerTest.TestExportRegistryKey() },

{ "Exit", () => { Console.WriteLine("Exiting the program"); Environment.Exit(0); } }

};

var menu = new Menu(items);

while (true)

{

menu.DisplayOptions();

string? option = menu.ChooseOption();

menu.HandleInput(option);

menu.WaitNextIteration();

}

// Кінець файлу

using Microsoft.Win32;

using System.Diagnostics;

// Ім'я файлу: RegManager.cs

// Ремарка: RegManager - клас, що включає логіку взаємодії з реєстром Windows

// Автор: Андрій Сахно

namespace ConsoleAppLab4.Logic

{

public class RegManager

{

// Програмно, вивести список усіх програм та служб які завантажуються

// автоматично для усіх користувачів та поточного користувача.

public void ListStartupPrograms(bool forAllUsers)

{

Console.WriteLine($"~ Listing startup programs for {(forAllUsers ? "all users" : "current user")}:");

RegistryKey baseKey;

if (forAllUsers)

baseKey = Registry.LocalMachine;

else

baseKey = Registry.CurrentUser;

using (RegistryKey key = baseKey.OpenSubKey(@"SOFTWARE\Microsoft\Windows\CurrentVersion\Run", false))

{

if (key != null)

{

foreach (string subkeyName in key.GetValueNames())

{

object? keyValue = key.GetValue(subkeyName);

if (keyValue != null)

{

string value = keyValue.ToString()!;

Console.WriteLine($"Program: {subkeyName}, Path: {value}");

}

}

}

}

}

// Додати програмно до автозавантаження програм для поточного

// користувача завантаження програми WinWord або іншої.

public void AddProgramToCurrentUserStartup(string programName, string programPath)

{

try

{

RegistryKey key = Registry.CurrentUser.OpenSubKey(@"SOFTWARE\Microsoft\Windows\CurrentVersion\Run", true);

if (key != null)

{

key.SetValue(programName, programPath);

Console.WriteLine($"Program '{programName}' added to current user's startup.");

}

else

{

Console.WriteLine("Error accessing registry key.");

}

}

catch (Exception ex)

{

Console.WriteLine($"An error occurred: {ex.Message}");

}

}

// Зробити програмно копію будь якого розділу реєстру

// у файл відповідного формату .reg

public void ExportRegistryKey(string key, string outputFile)

{

Process.Start("regedit.exe", $"/E {outputFile} {key}");

}

}

}

// Кінець файлу

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

// Ім'я файлу: Program.cs

// Ремарка: RegManagerTest використовує клас RegManager та створений

// для його тестування та передачі методів як Action делегатів

// у елементи меню

// Автор: Андрій Сахно

namespace ConsoleAppLab4.Logic

{

public class RegManagerTest

{

private readonly RegManager \_regManager;

public RegManagerTest()

{

\_regManager = new RegManager();

}

public void ListStartupProgramsAllUsers()

{

\_regManager.ListStartupPrograms(true);

}

public void ListStartupProgramsCurrentUser()

{

\_regManager.ListStartupPrograms(false);

}

public void AddSpotifyToCurrentUserStartup()

{

string programName = "Spotify";

string programPath = @"C:\Users\User\AppData\Roaming\Spotify\Spotify.exe";

\_regManager.AddProgramToCurrentUserStartup(programName, programPath);

}

public void TestExportRegistryKey()

{

string key = @"HKEY\_CURRENT\_USER\SOFTWARE\TestLab4";

string outputFilePath = "C:\\Users\\User\\Desktop\\key.reg";

try

{

\_regManager.ExportRegistryKey(key, outputFilePath);

Console.WriteLine($"Registry key exported successfully: {outputFilePath}");

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

}

}

}

// Кінець файлу

using Microsoft.Win32.TaskScheduler;

using System.Threading.Tasks;

// Ім'я файлу: TaskServiceManager.cs

// Ремарка: TaskServiceManager надає інформацію про задачі, зареєстровані

// у планувальнику задач

// Автор: Андрій Сахно

namespace ConsoleAppLab4.Logic

{

public class TaskServiceManager

{

// Вивести список усіх завдань, які зареєстровані у планувальнику

// задач системи. Інформації отримати з відповідного розділу реєстру,

// як для усіх користувачів так і для поточного користувача.

public void DisplayAllUserTasks()

{

using (TaskService ts = new TaskService())

{

foreach (Microsoft.Win32.TaskScheduler.Task task in ts.AllTasks)

{

DisplayTask(task);

}

}

}

public void DisplayCurrentUserTasks()

{

using (TaskService ts = new TaskService())

{

string userSID = System.Security.Principal.WindowsIdentity.GetCurrent().User.Value;

foreach (Microsoft.Win32.TaskScheduler.Task task in ts.FindAllTasks(t => t.Definition.Principal.UserId == userSID || t.Definition.Principal.UserId == null))

{

DisplayTask(task);

}

}

}

private void DisplayTask(Microsoft.Win32.TaskScheduler.Task task)

{

Console.WriteLine("Task Name: " + task.Name);

Console.WriteLine("Task Path: " + task.Path);

Console.WriteLine("Task State: " + task.State);

Console.WriteLine("Task Last Run Time: " + task.LastRunTime);

Console.WriteLine("Task Next Run Time: " + task.NextRunTime);

Console.WriteLine("---------------------------------------");

}

}

}

// Кінець файлу

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

// Ім'я файлу: Menu.cs

// Ремарка: Menu відображає консольне меню опцій, обробляє вибір користувача та виконує відповідні дії.

// Автор: Андрій Сахно

namespace ConsoleAppLab4.Logic

{

class Menu

{

private Dictionary<string, Action> menuItems;

public Menu(Dictionary<string, Action> items)

{

menuItems = items;

}

public void DisplayOptions()

{

Console.Clear();

Console.WriteLine("\n\n================ Menu ================\n");

int i = 1;

foreach (var item in menuItems)

{

Console.WriteLine($" Option {i}: {item.Key}");

i++;

}

Console.WriteLine("\n=======================================");

}

public string? ChooseOption()

{

Console.Write("\nYour choice ==> ");

string? userInput = Console.ReadLine();

Console.WriteLine();

return userInput;

}

public void HandleInput(string? userInput)

{

try

{

int operationNumber = int.Parse(userInput!);

menuItems.ElementAt(operationNumber - 1).Value.Invoke();

}

catch (Exception ex)

{

Console.WriteLine($"{ex.Message}\nPlease, try again.");

}

}

public void WaitNextIteration()

{

Console.Write("\nPress any key to continue.");

Console.ReadLine();

}

}

}

// Кінець файлу

***Висновок:*** У результаті виконання лабораторної роботи було розроблено програму "ConsoleAppLab4", яка забезпечує користувачеві зручні інструменти для управління автозавантаженням програм та відслідковуванням задач у планувальнику системи. Використовуючи структурований підхід до програмування та ефективне використання бібліотек Windows, програма дозволяє користувачеві з легкістю контролювати певну роботу своєї системи.