**ЛАБОРАТОРНА РОБОТА № 14**

**Варіант 2**

**Тема:** Робота з файлами у мові C#

***Мета*** : навчитися застосовувати бібліотеки мови C# для виконання операцій над текстовими та двійковими файлами; навчитися використовувати серіалізацію та десеріалізацію об’єктів.

**Хід роботи:**

**Завдання**:

**Лістинг Form1**:

using ClassLibrary;

using System;

using System.IO;

using System.Linq;

using System.Windows.Forms;

namespace WindowsFormsApp

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void archiveToolStripMenuItem1\_Click(object sender, EventArgs e)

{

var dialog = new SaveFileDialog

{

AddExtension = true,

Filter = "packed files (\*.gzar)|\*.gzar"

};

var result = dialog.ShowDialog();

if (result != DialogResult.OK)

{

return;

}

Packing.Pack(dialog.FileName, listBox1.SelectedItems.Cast<FileEntry>());

}

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

{

var dialog = new OpenFileDialog

{

Multiselect = true

};

var result = dialog.ShowDialog();

if (result != DialogResult.OK)

{

return;

}

listBox1.Items.Clear();

foreach (var file in dialog.FileNames)

{

var fileEntry = new FileEntry(file);

listBox1.Items.Add(fileEntry);

}

}

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

{

listBox1.Items.Clear();

}

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

{

var dialog = new FolderBrowserDialog();

var result = dialog.ShowDialog();

if (result != DialogResult.OK)

{

return;

}

Packing.Unpack(dialog.SelectedPath, listBox1.SelectedItems.Cast<FileEntry>().Where(file => file.IsPacked()));

}

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

{

var dialog = new FolderBrowserDialog();

var result = dialog.ShowDialog();

if (result != DialogResult.OK)

{

return;

}

Crypt.Encrypt(dialog.SelectedPath, listBox1.SelectedItems.Cast<FileEntry>(), Crypt.Key);

}

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

{

var dialog = new FolderBrowserDialog();

var result = dialog.ShowDialog();

if (result != DialogResult.OK)

{

return;

}

Crypt.Decrypt(dialog.SelectedPath,

listBox1.SelectedItems.Cast<FileEntry>().Where(file => file.IsEncrypted()), Crypt.Key);

}

private void listBoxFile\_DragEnter(object sender, DragEventArgs e)

{

if (e.Data.GetDataPresent(DataFormats.FileDrop) == false)

{

return;

}

e.Effect = DragDropEffects.Copy;

}

private void listBoxFile\_DragDrop(object sender, DragEventArgs e)

{

string[] files = (string[])e.Data.GetData(DataFormats.FileDrop);

string path = Path.GetDirectoryName(files.FirstOrDefault());

foreach (var file in files)

{

if (FileUtils.IsDirectory(file))

{

foreach (var subFile in Directory.EnumerateFiles(file, "\*.\*", SearchOption.AllDirectories))

{

listBox1.Items.Add(new FileEntry(FileUtils.GetRelativePath(subFile, path), subFile));

}

}

else

{

listBox1.Items.Add(new FileEntry(Path.GetFileName(file), file));

}

}

}

private void Form1\_Load(object sender, EventArgs e)

{

}

}

}

**Лістинг Crypt**:

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ClassLibrary

{

public class Crypt

{

public const int Key = 234232589;

public static void Encrypt(string path, IEnumerable<FileEntry> files, int key)

{

foreach (var file in files)

{

Encrypt(path, file, key);

}

}

private static void Encrypt(string path, FileEntry file, int key)

{

var encr = Getencr(file, key);

using (var stream = new FileStream(Path.Combine(path, file.Name + ".crypt"), FileMode.Create))

{

stream.Write(encr, 0, encr.Length);

}

}

public static void Decrypt(string path, IEnumerable<FileEntry> files, int key)

{

foreach (var file in files)

{

Decrypt(path, file, key);

}

}

private static void Decrypt(string path, FileEntry file, int key)

{

var decr = Getdecr(file, key);

using (var stream = new FileStream(Path.Combine(path, Path.GetFileNameWithoutExtension(file.Name)),

FileMode.Create))

{

stream.Write(decr, 0, decr.Length);

}

}

private static byte[] Getencr(FileEntry file, int key)

{

byte[] con = file.GetBytes();

byte[] encr = new byte[con.Length];

for (int i = 0; i < con.Length; i++)

{

encr[i] = (byte)(con[i] ^ key);

}

return encr;

}

private static byte[] Getdecr(FileEntry file, int key)

{

return Getencr(file, key);

}

}

}

**Лістинг FileEntry**:

namespace ClassLibrary

{

public class FileEntry

{

public string Name { get; }

public string Path { get; }

public FileEntry(string path)

{

Name = System.IO.Path.GetFileName(path);

Path = path;

}

public FileEntry(string name, string path)

{

Name = name;

Path = path;

}

public byte[] GetBytes() => System.IO.File.ReadAllBytes(Path);

public string GetText() => System.IO.File.ReadAllText(Path);

public bool IsPacked() => System.IO.Path.GetExtension(Name) == ".gzar";

public bool IsEncrypted() => System.IO.Path.GetExtension(Name) == ".crypt";

}

}

**Лістинг FileUtils**:

using System;

using System.IO;

namespace ClassLibrary

{

public class FileUtils

{

public static bool IsDirectory(string path)

{

return File.GetAttributes(path).HasFlag(FileAttributes.Directory);

}

public static string GetRelativePath(string fullPath, string Path)

{

if (Path.EndsWith("\\") == false)

{

Path += "\\";

}

var baseUri = new Uri(Path);

var fullUri = new Uri(fullPath);

return baseUri.MakeRelativeUri(fullUri).ToString().Replace("/", "\\");

}

}

}

**Лістинг Packing**:

using System;

using System.Collections.Generic;

using System.IO;

using System.Text;

namespace ClassLibrary

{

public class Packing

{

public static void Pack(string path, IEnumerable<FileEntry> files)

{

using (var stream = new FileStream(path, FileMode.Create))

{

var root = Path.GetDirectoryName(path);

foreach (var file in files)

{

Pack(stream, file, root);

}

}

}

public static void Unpack(string path, IEnumerable<FileEntry> files)

{

foreach (var file in files)

{

Unpack(path, file.Path);

}

}

private static void Pack(Stream stream, FileEntry file, string root)

{

byte[] content = file.GetBytes();

byte[] fileName = Encoding.UTF8.GetBytes(file.Name);

stream.Write(BitConverter.GetBytes(fileName.Length), 0, sizeof(int));

stream.Write(fileName, 0, fileName.Length);

stream.Write(BitConverter.GetBytes(content.Length), 0, sizeof(int));

stream.Write(content, 0, content.Length);

}

private static void Unpack(string path, string packedFile)

{

using (var stream = new FileStream(packedFile, FileMode.Open))

{

while (stream.Position < stream.Length)

{

byte[] fileName = ReadBytes(stream);

byte[] content = ReadBytes(stream);

string combinedPath = Path.Combine(path, Encoding.UTF8.GetString(fileName));

string directoryPath = Path.GetDirectoryName(combinedPath);

if (Directory.Exists(directoryPath) == false)

{

Directory.CreateDirectory(directoryPath ?? string.Empty);

}

var unpackedStream = new FileStream(combinedPath, FileMode.Create);

unpackedStream.Write(content, 0, content.Length);

}

}

}

private static int ReadInt(Stream stream)

{

byte[] bytes = new byte[sizeof(int)];

\_ = stream.Read(bytes, 0, sizeof(int));

return BitConverter.ToInt32(bytes, 0);

}

private static byte[] ReadBytes(Stream stream)

{

int size = ReadInt(stream);

byte[] bytes = new byte[size];

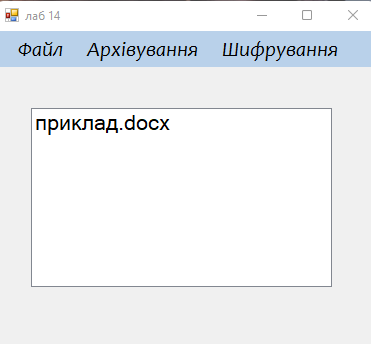
\_ = stream.Read(bytes, 0, size);

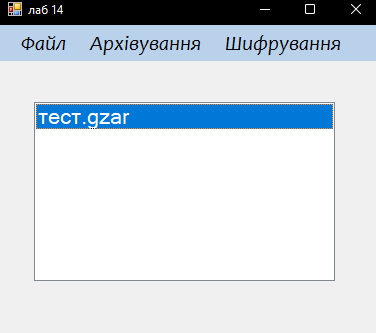
return bytes;

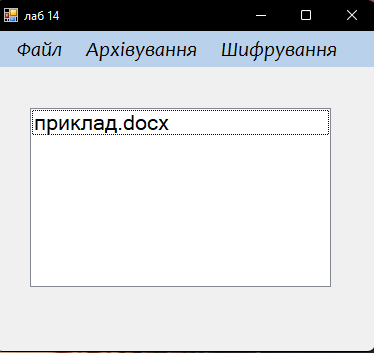
}

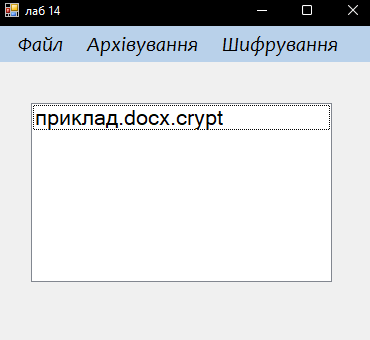
}

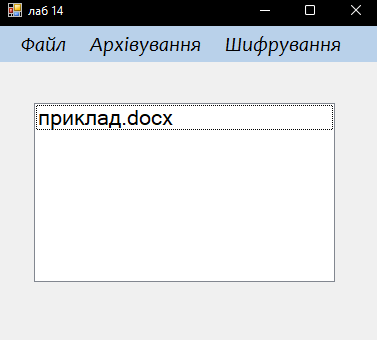
}

****

****

****

****

****

***Висновки:*** я навчився застосовувати бібліотеки мови C# для виконання операцій над текстовими та двійковими файлами; навчився використовувати серіалізацію та десеріалізацію об’єктів.