Reading and writing files

2025-01-06

I. Introduction

.Net offers several ways to work with files. You can use a FileStream for this purpose. However, there's also a helper class `System.IO.File` that can read/write files completely in one go.

I.1. Reading files

The File class provides several methods for reading files:

ReadAllText: Reads the entire content of a file as a string

```
string content = File.ReadAllText("TestFile.txt");
```

ReadAllLines: Reads all lines of a file into a string array

```
string[] lines = File.ReadAllLines("TestFile.txt");
foreach (string line in lines)
{
    Console.WriteLine(line);
}
```

ReadAllBytes: Reads the content of a file as a byte array

```
byte[] bytes = File.ReadAllBytes("TestFile.txt");
```

I.2. Writing files

The File class also offers several methods for writing to files: WriteAllText: Writes a string to a file

```
File.WriteAllText("output.txt", "Hallo Welt!");
```

WriteAllLines: Writes a string array line by line to a file

```
string[] lines = { "Erste Zeile", "Zweite Zeile", "Dritte Zeile" };
File.WriteAllLines("output.txt", lines);
```

WriteAllBytes: Writes a byte array to a file

```
byte[] bytes = { 72, 101, 108, 108, 111 }; // "Hello" in ASCII
File.WriteAllBytes("output.bin", bytes);
```

II. Excercises

1. Implement a simple logger that writes messages to a file and adds the date of each message.

- Hint: You can get the current date with: System.DateTime.Now

Reading and writing files

2. Implement a program that manages a “highscore” list. Users can enter their names and scores, which are then saved in a file. When the program starts, the current highscore list should be displayed.