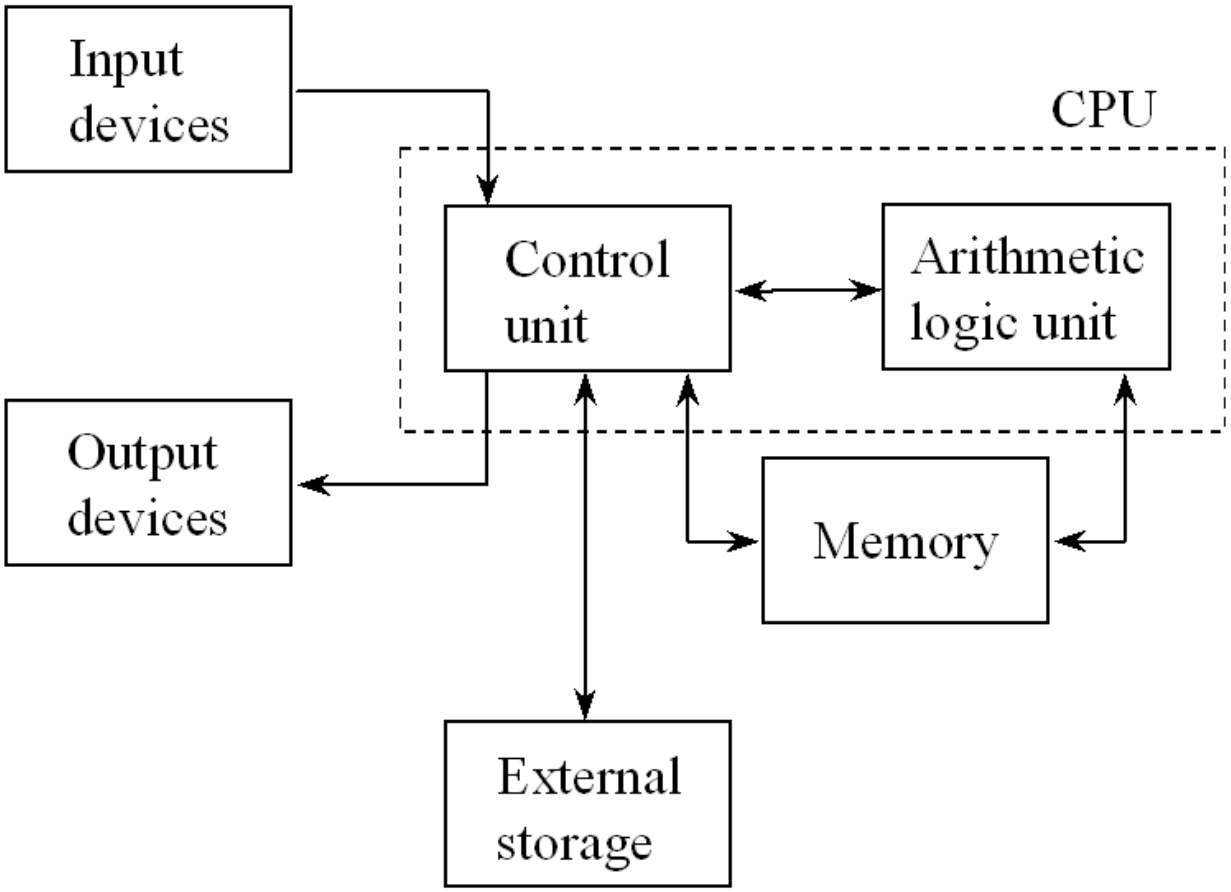
10.Вход и изход

Съдържание

- Потоци
- Четци и писачи
- Файлова система

Предистория



Какво са потоците

- Абстракция, която позволява извършане на операции за вход и изход в нашата програма
- Виртуален канал, който свързва нашата програма с входно-изходно устройство
- Подредена серия от байтове
- Последователен достъп
- Отложен достъп

Потоците в .NET Framework

using System.IO

Класификация

- Базови потоци – пишат и четат директно от външен механизъм (файл, паметта, мрежови ресурс)
- Преходни потоци – пишат и четат от друг поток, като добавят функционалност (буфериране, криптиране)

Класът System.IO.Stream

- Абстрактен базов клас за всички потоци
- Дефинира основните операции с потоци
  - конструиране
  - четене
  - писане
  - позициониране
  - затваряне

Основни членове на класа Stream

- Methods
  - Read / ReadByte
  - Write / WriteByte
  - Seek
  - Flush
  - Close
- Properties
  - Length
  - Position
  - CanRead
  - CanWrite
  - CanSeek

Забележка: не всеки поток позволява извършване на всички операции.

Основни потоци

- Базови
  - System.IO.FileStream
  - System.IO.MemoryStream
  - System.Net.Sockets.NetworkStream
- Преходни
  - System.IO.BufferedStream
  - System.IO.Compression.DeflateStream
  - System.Security.Cryptography.CryptoStream

Пример

```
char[] s = Encoding.UTF8.GetBytes("Hello World.");
Stream file = new FileStream("file.txt", FileMode.Create);
file.Write(s, 0, s.Length);
file.Close();
```

Не

Важно: потоците използват “неуправлявани” ресурси – винаги ги затваряйте

По-добре

```
char[] s = Encoding.UTF8.GetBytes("Hello World.");
Stream file = new FileStream("file.txt", FileMode.Create);
try
{
    file.Write(s, 0, s.Length);
}
finally
{
    file.Close();
}
```

Да

Важно: потоците са потенциален източник на изключения – винаги ги поставяйте в try-finally блок

Още по-добре

```
using (Stream file = new FileStream("file.txt", FileMode.Create))
{
    char[] s = Encoding.UTF8.GetBytes("Hello World.“);
    file.Write(s, 0, s.Length);
}
```

Интерфейсът System.IDisposable

- Дефинира метода void Dispose()
- Употреба – за освобождаване на неуправлявани ресурси
- using конструкцията и IDisposable

```
using (IDisposable file = new …)
{
    // Your code here
}
```

```
IDisposable file = new …
try
{
    // Your code here
}
finally
{
    file.Dispose();
}
```

Dispose vs. Finalize

- Finalize()
  - Предназначение – освобождаване на неуправлявани ресурси
  - Вика се от GC
  - Вика се в неопределен момент
- Dispose()
  - Предназначение – освобождаване на неуправлявани ресурси
  - Вика се от програмиста
  - Вика се в определен момент

Demo

Четци и писачи

- Надграждат потоците с допълнителна функционалност
- Улесняват работата с потоците

Видове четци и писачи

- Двоични
  - Служат за четене и писане на структури
  - System.IO.BinaryReader & System.IO.BinaryWriter
- Текстови
  - Служат за четене и писане на текст
  - System.IO.TextReader - абстрактен
    - System.IO.StreamReader
    - System.IO.StringReader
  - System.IO.TextWriter – абстрактен
    - System.IO.StreamWriter
    - System.IO.StringWriter

Demo

Четци, писачи и кодиране

- Данните винаги се съхраняват в двоичен формат, като се използва някакво кодиране
- Когато четем трябва да знаем в какъв формат са кодирани данните при записване
- Когато пишен трябва да знаем в какъв формат ще бъдат четени данните
- Кодирането по подразбиране е UTF8
- В конструкторите на четците и писачите можем да окажем различно кодиране

Файлова система

- Компонента на ОС,предназначена да управлява постоянни обекти

- Постоянни обекти – обектите, които съществуват по-дълго от процесите, които ги създават и използват

Файловата система и.NET Framework

using System.IO

Основни класове

- Файлове
  - File
  - FileInfo
- Директории
  - Directory
  - DirectoryInfo
- Атрибути
  - FileAttributes

File & Directory vs. FileInfo & DirectoryInfo

- File & Directory
  - Предоставят статични методи за работа с ФС
  - Използват се за извършване на еденична операция над конкретен обект от ФС
- FileInfo & DirectoryInfo
  - Предоставят методи за работа с ФС през инстанция
  - Използват се за извършване на множество операции над конкретен обект от ФС

Особености

- FileInfo, DirectoryInfo и методът Refresh()
  - Синхронизация м/у данните на ФС и данните в текущата инстанция
  - Не е автоматична, а грижа на програмиста
- Delete() методите
  - Не местят файла/директорията в Recycle Bin-а
- Directory.GetCurrentDirectory() – връща текущата директория

Класът System.IO.Path

- Предоставя статични методи за манипулация на стрингови пътища във ФС
- Не работи със самата ФС
- Не валидира съществуването на обекти от ФС
- Валидира самите пътища за невалидни символи или семантика

Класът FileSystemWatcher

- Предоставя функционалност за наблюдаване на промените в директория
- Може да се специфицира филтър за кои файлове да бъдат следени
- Може да се специфицира филтър за кои промени да бъдат следени
- Предоставя събития, които сигнализират за различните промени

Demo

Задача 1

Напишете програма, която прави копие на файл като обръща реда на думито във файла. Т.е. първата дума става последна, втората предпоследна и т.н.

Задачи 2

Имплментирайте ваш собствен текстов четец и писач, който заменя малките букви с големи и големите с малки

Задача 3

Напишете програма, която работи от командния ред и извършва операции с файлове. Нека да може да архивира файлове, да криптира файлове и да криптира и архивира едновременно. Също така да поддържа и обратните операции. Параметрите, които да приема да са име на файл и ключове за това каква операция да се изпълни.

Пример:

cryptoarch.exe .\Docs\doc1.docx \arch [\encrypt]

Задача 4

Напишете програма, която търси даден стринг във всички текстови (*.txt) файлове в дадена директория и поддиректориите и. Резултатът да се запише в Result.txt. За всеки намерен файл да се запише пълния път до него и съдържанието на първия ред, който съдържа търсения стринг.

Въпроси?