Programování

Práce se souborem

Vstupní a výstupní proudy doposud

- **Konzolové aplikace**
  - *Console.ReadLine()*
  - *Console.WriteLine()*
  - Vstup je vždy v podobě řetězce a je nutné jej zpracovat
  - Pokud je očekáván vstup, aplikace bez vložení vstupu nepokračuje
- **Formulářové aplikace**
  - Jednotlivé komponenty mají své vlastnosti (text, value, colection,…)
  - Datové typy vstupů a výstupů se odvíjí od použité komponenty
  - Aplikace může skončit v chybovém stavu, protože nemusí být uveden vstup

Práce se souborem

- Knihovna **Systém.IO**

```
using System.IO;
```

- Pro čtení a zápis do souboru je nutné aktivovat a inicializovat datové proudy **stream**
- Pro práci se souborem využíváme třídy:
  - File
  - FileStream
  - StreamWriter
  - StreamReader

Typy souborů

- Na soubory můžeme koukat dvěma způsoby:
  - **Binární**
    - Veřejné (.png, .jpg, …)
    - Proprietární (.docx, …)
  - **Textové**
    - Strukturované (.csv, .xml, …)
    - Nestrukturované (.txt)
- Název souboru **path** se skládá nejen pojmenování souboru, ale rovněž cesty k souboru včetně disku, kde je soubor uložen

Postup práce se souborem

- Práce se souborem se sestává ze tří kroků:
  1. Vytvoření proudu pro čtení/zápis
  2. Práce se samostatným souborem
  3. Uzavření proudu
- Uzavření proudu je nutné pro uvolnění zdrojů, které si proud nárokuje
- V případě vyšších programovacích jazyků jsou neuvolněné zdroje po ukončení likvidovány pomocí **GARBAGE COLLECTOR** v nižších způsobuje úniky paměti – **MEMORY LEAKS**

Ukázka čtení textového souboru

- Proměnná path obsahuje cestu ke konkrétnímu souboru
- Klíčovým slovem **using** uvozujeme založení proudu, který budeme využívat
  - Moment, kdy používáme proud je ohraničen blokem kódu
  - Použitím using zajistíme automatické zavolání funkce **Dispose()**
- Na konzoli vypisujeme řádek po řádku dokud nenarazíme na konec souboru
- Po ukončení práce proud uzavíráme

```
string path = "C:\\soubor.txt";

using (StreamReader sr = new StreamReader(path))
{
    while (!sr.EndOfStream)
    {
        Console.WriteLine(sr.ReadLine());
    }
    sr.Close();
}
```

Vlastnost třídy StreamReader

Uzavírání souborových proudů

- Pro uzavření využíváme funkcí Close() a Dispose()
- **Funkce Close()**
  - Uzavírá konkrétní souborový proud, ale ponechává jej alokovaný pro jeho opětovné použití
  - V případě, že se při práci s proudem objeví výjimka, proud může zůstat neuzavřen
- **Funkce Dispose()**
  - Uzavírá všechny momentálně nepoužívané proudy
  - V kombinaci s využitím bloku uvozeným using zajistí uzavření proudů i v případě výjimky

Problém s neexistujícím souborem

- V ukázce čtení ze souboru může vzniknout výjimka v případě, že daný soubor nebude existovat
- **FileNotFoundException**
- Práce se souborem je jedna z nejkritičtějších částí libovolného programu
- Možná řešení:
  - Využití již známe konstrukce **try-catch**
  - Využití funkce ze třídy File – **Exists()**
- Obě varianty vytváří validní řešení, ale efektivnější je použií metody než pomalejšího try-catch

Ukázka zápisu do souboru

- Postup pro zápis do souboru je obdobný jako čtení ze souboru
- Před zápisem je nutné prvně ustanovit vstupní proud **StreamWriter**
- Při konstrukci vstupního proudu se můžeme rozhodnout, zda budeme přidávat na konec souboru nebo soubor přepíšeme
- V případě, že soubor neexistuje, je automaticky vytvořen

```
string path = "C:\\soubor.txt";

using (StreamWriter sw = new StreamWriter(path))
{
    sw.WriteLine("ahoj");
    sw.WriteLine("světe");
    sw.Close();
}
```

```
string path = "C:\\soubor.txt";

using (StreamWriter sw = new StreamWriter(path,true))
{
    sw.WriteLine("ahoj");
    sw.WriteLine("světe");
    sw.Close();
}
```