**Soubory a serializace**

**Soubory**

Data, které jsou používána při běhu programu na stacku a heapu jsou v OP a po vypnutí programu se ztratí.

Pokud se chcem, aby data byla trvalá (perzintentní) a bylo je možné při opětovném zapnutí programu znova načíst, tak se musí ukládat mimo zdrojový kód, třeba do souborů nebo do databáze.

K souborům je přístup delší než k datům v OP.

Soubory se rozdělují na:

* Textové soubory s plochou strukturou (txt, csv)
* Textové soubory s vnitřní hierarchií (xml, json)
* Soubory binárního typu
* Databáze

Windows zakuje zápis na systémový oddíl disku, musí být povolen administrátorem nebo se nejedná o zápis do složek uživatele.

Také antivirový program může vyhodit jako škodlivý a také ho zablokuje.

Stream = datový proud, přes tento proud lze číst a zapisovat data.

K práci se soubory je využívá namespace System.IO

IOException obsahuje výjimka na práci se soubory, které by bylo dobré ohlídat: DirectoryNotFoundException, DriveNotFoundException, FileLoadException, FileNotFoundException.

**K práci se soubory se používají třídy:**

File

Poskytuje statické metody pro vytváření, kopírování, odstraňování, přesouvání a otevírání jednoho souboru a pomůcky při vytváření FileStream objektů.

FileInfo

Poskytuje vlastnosti a metody instance pro vytváření, kopírování, odstraňování, přesouvání a otevírání souborů a pomůcky při vytváření FileStream objektů. Tuto třídu nelze zdědit.

FileStream

Poskytuje Stream soubor, který podporuje synchronní i asynchronní operace čtení i zápisu.

Directory

Zpřístupňuje statické metody pro vytváření, přesouvání a vytváření výčtu adresářů a podadresářů. Tuto třídu nelze zdědit.

DirectoryInfo

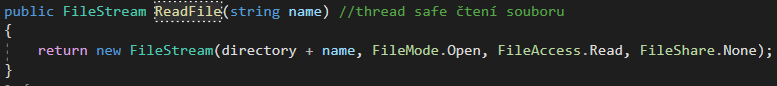
Zpřístupňuje metody instance pro vytváření, přesouvání a vytváření výčtu adresářů a podadresářů. Tuto třídu nelze zdědit.

Path

Provádí operace s String instancemi, které obsahují informace o cestě k souboru nebo adresáři. Tyto operace se provádějí na různých platformách.

\*\*příklad

Čtení ze souborů je možné thread-safe, tím pádem může číst soubor více vláken, ale zápis jednoduše nelze udělat (můžou se použít lockery, nebo thread-safe kolekce, která by sloužila jako dočasná úschovna dat, které by měla být zapsána a jedno vlákno by z ní zapisovalo)



Nejlehčí zápis a čtení do souborů je přes StreamReader a StreamWriter. Writer obsahuje metody pro zápis jako Write(), WriteLine() a také metodu Flush(pro zapsání dat z bufferu a vymazání bufferu, kde jsou data k zápisu dočasně uschována). Reader obsahuje metody pro čtení jako ReadLine(), ReadToEnd().

File může zastát nějaké funkce StreamWriteru a StreamReaderu, ale ne všechny. Napřiklad metoda ReadAllLines(path k souboru) vrátí pole všech dat v souboru, metoda WriteAllLines(pole stringů, path k souboru) nebo pro přidání dat do souboru AppendAllLines

**Serializace**

Pomáhá při uchování objektů do souborů, databáze,…

Převede objekt z OP na streamy bytů a ty jsou uloženy do nějakého souboru.

**Deserializace**

Je to opak serializace, převede data ze souborů na objekty v OP.

Je to užitečné při přenosů dat jako soubory mezi počítači, třeba po síti. Nebo při ukládání nastavení aplikace, které by bylo vytvořeno objektově a chtělo by se uchovávat do dalšího spuštění aplikace.

Jsou možnosti jak serializovat objekty, do binárního typu, ty nelze jednoduše číst anebo do CSV, XML nebo JSON

Objekt, který chcem serializovat musí mít konstruktor bez vstupních parametrů (můžou být 2 jeden normální s parametry a druhý prázdný pro serializaci), protože deserializer nejdříve vytvoří novou instanci a poté jí nastavuje vlastnosti jak je čte ze souboru.